Technical Document **2963**
May 1997

Terrain Parabolic Equation Model (TPEM) Computer Software Configuration Item (CSCI) Documents

A. E. Barrios

Approved for public release; distribution is unlimited.

Naval Command, Control and Ocean Surveillance Center
RDT&E Division, San Diego, CA 92152–5001

Technical Document **2963**

May 1997

Terrain Parabolic Equation Model (TPEM) Computer Software Configuration Item (CSCI) Documents

A. E. Barrios

Approved for public release; distribution is unlimited.

Naval Command, Control and Ocean Surveillance Center
RDT&E Division, San Diego, CA 92152–5001

NAVAL COMMAND, CONTROL AND OCEAN SURVEILLANCE CENTER RDT&E DIVISION
San Diego, California 92152–5001

H. A. WILLIAMS, CAPT, USN	**R. C. KOLB**
Commanding Officer	**Executive Director**

ADMINISTRATIVE INFORMATION

The work detailed in this report was performed for the Space and Naval Warfare Systems Command (PMW–185) by the Naval Command, Control and Ocean Surveillance Center RDT&E Division, Tropospheric Branch, Code D883.

Released by	Under authority of
R. A. Paulus, Head	J. H. Richter, Head
Tropospheric Branch	Propagation Division

SB

NAVAL COMMAND, CONTROL AND OCEAN SURVEILLANCE CENTER
RESEARCH, DEVELOPMENT, TEST & EVALUATION DIVISION
SAN DIEGO, CALIFORNIA 92152–5001

10 July 1997

Technical Document 2963
Terrain Parabolic Equation Model (TPEM) Computer Software Configuration Item (CSCI) Documents
By: A. E. Barrios
Dated: May 1997

Literature Change

1. Replace the existing cover and title page with the attached corrected pages.
2. Replace the existing Form 298 with the attached corrected page.

SOFTWARE REQUIREMENTS SPECIFICATION

FOR THE

TERRAIN PARABOLIC EQUATION MODEL CSCI

May 1, 1997

Prepared for:

Space and Naval Warfare Systems Command (PMW-185)
Washington, DC

and

Naval Sea Systems Command (PEO USW ASTO-E/F)
Washington, DC

Prepared by:

Naval Command, Control and Ocean Surveillance Center
Research, Development, Test and Evaluation Division
Tropospheric Branch (Code D883)
San Diego, CA 92152-738

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. SCOPE

1.1 Identification.

Terrain Parabolic Equation Model (TPEM) computer software configuration item (CSCI). The purpose of the TPEM CSCI is to calculate range-dependent electromagnetic (EM) system propagation loss within a heterogeneous atmospheric medium over variable terrain, where the radio-frequency index of refraction is allowed to vary both vertically and horizontally, also accounting for terrain effects along the path of propagation.

1.2 System Overview

The TPEM CSCI model will calculate propagation loss values as electromagnetic (EM) energy propagates through a laterally heterogeneous atmospheric medium where the index of refraction is allowed to vary both vertically and horizontally, also accounting for terrain effects along the path of propagation. Numerous Tactical Environmental Support System (TESS) applications require EM-system propagation loss values. The required TPEM model described by this document may be applied to two such TESS applications, one which displays propagation loss on a range versus height scale (commonly referred to as a coverage diagram) and one which displays propagation loss on a propagation loss versus range/height scale (commonly referred to as a loss diagram).

1.3 Document Overview

This document specifies the functional requirements that are to be met by the TPEM CSCI. A discussion of the input software requirements is presented together with a general description of the internal structure of the TPEM CSCI as it relates to the CSCI's capability.

2. REFERENCED DOCUMENTS

(a) Bergland, G. D., "A Radix-eight Fast Fourier Transform Subroutine for Real-valued Series," IEEE Trans. Audio and Electro-acoust., Vol. AU-17, pp. 138-144, 1969.

(b) Cooley, J. W., P. A. W. Lewis and P. D. Welsh, "The Fast Fourier Transform Algorithm: Programming Considerations in the Calculation of Sine, Cosine and Laplace Transforms," J. Sound Vib., Vol. 12, pp. 315-337, 1970.

(c) Tappert, F. D., "The Parabolic Approximation Method," Wave Propagation and Underwater Acoustics, J. B. Keller and J. S. Papadakis, Eds., New York, Springer-Verlag, pp. 224-285, 1977.

(d) Commander-In-Chief, Pacific Fleet Meteorological Requirement (PAC MET) 87-04, "Range Dependent Electromagnetic Propagation Models," 1987.

(e) Dockery, G. D., "Modeling Electromagnetic Wave Propagation in the Troposphere Using the Parabolic Equation", IEEE Trans. Antennas Propagat., Vol. 36, No. 10, pp. 1464-1470, October 1988.

(f) Naval Oceanographic Office, "Software Documentation Standards and Coding Requirements for Environmental System Product Development," April 1990.

(g) Kuttler, J. R. and G. D. Dockery., "Theoretical Description of the Parabolic Approximation/Fourier Split-Step Method of Representing Electromagnetic Propagation in the Troposphere," Radio Sci., Vol. 26, pp. 381-393, 1991.

(h) American National Standards Institute (ANSI), "Program Language - Fortran - Extended", 1992.

(i) Barrios, A. E., "Terrain and Refractivity Effects on Non-Optical Paths," AGARD Conference Proceedings 543, Multiple Mechanism Propagation Paths (MMPPs): Their Characteristics and Influence on System Design, pp. 10-1 to 10-9, October 1993.

(j) Barrios, A. E., "A Terrain Parabolic Equation Model for Propagation in the Troposphere," IEEE Trans. Antennas Propagat., Vol. 42, pp. 90-98, January 1994.

(k)Naval Oceanographic Office, "Software Documentation Standards for Environmental System Product Development," February 1996.

(l) Barrios, A. E., "Terrain Parabolic Equation Model (TPEM) Version 1.5 User's Manual," Naval Command, Control and Ocean Surveillance Center, RDT&E Division, San Diego, CA, NRaD TD 2898, February 1996.

(m)International Radio Consulting Committee (CCIR). 1986. "Propagation in Non-Ionized Media," Recommendations and Reports of the CCIR, vol. V.

3. REQUIREMENTS

3.1 CSCI Capability Requirements

The required TPEM CSCI propagation model is a pure split-step parabolic equation (PE) model that allows for range-dependent refractivity and variable terrain along the path of propagation. It should calculate propagation loss both in range and altitude.

The TPEM CSCI should provide propagation loss for moderately low angles and heights. It is not required to provide propagation loss for *all* heights and ranges desired. Propagation loss values will be provided at all heights from *at least* 90% of the desired maximum range to the maximum range. The TPEM CSCI should allow for horizontal and vertical antenna polarization, finite conductivity based on user-specified ground composition and dielectric parameters, and the complete range of EM system parameters and most antenna patterns required by TESS.

The program flow of the required TPEM CSCI is illustrated in Figure 3-1. Note that the TPEM CSCI is shown within the context of a calling CSCI application such as one that generates a coverage or loss diagram. The efficient implementation of the TPEM CSCI will have far reaching consequences upon the design of an application CSCI beyond those mentioned in Section 3.10. For example, Figure 3-1 shows checking for the existence of a previously created TPEM output file prior to the access of the TPEM CSCI. The application CSCI will have to consider if the atmospheric or terrain environment has changed since the TPEM output file was created or if any new height or range requirement is accommodated within the existing TPEM CSCI output file. Because these and many more considerations are beyond the scope of this document to describe, an application CSCI designer should work closely with the TPEM CSCI development agency in the implementation of the TPEM CSCI. Figures 3-2 and 3-3 illustrate the program flow for the main compute software components (CSC), PEINIT CSC and the PESTEP CSC, respectively.

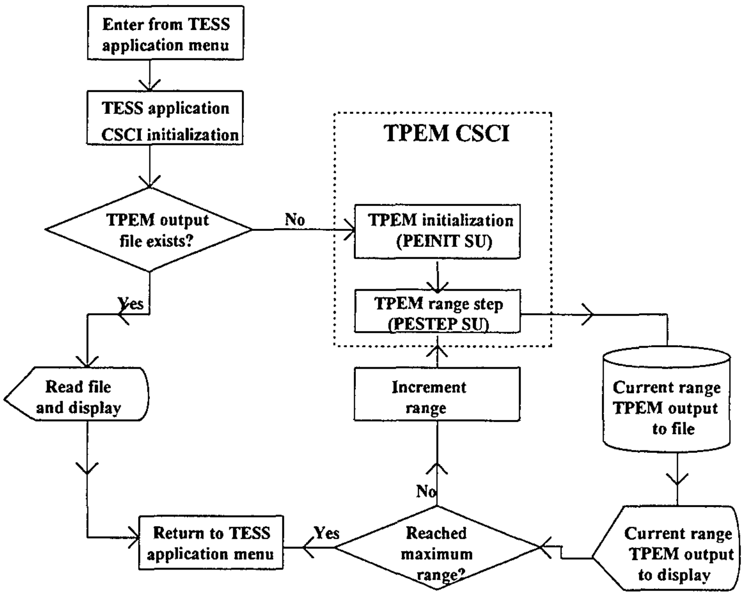

Figure 3-1 Program flow of the TPEM CSCI

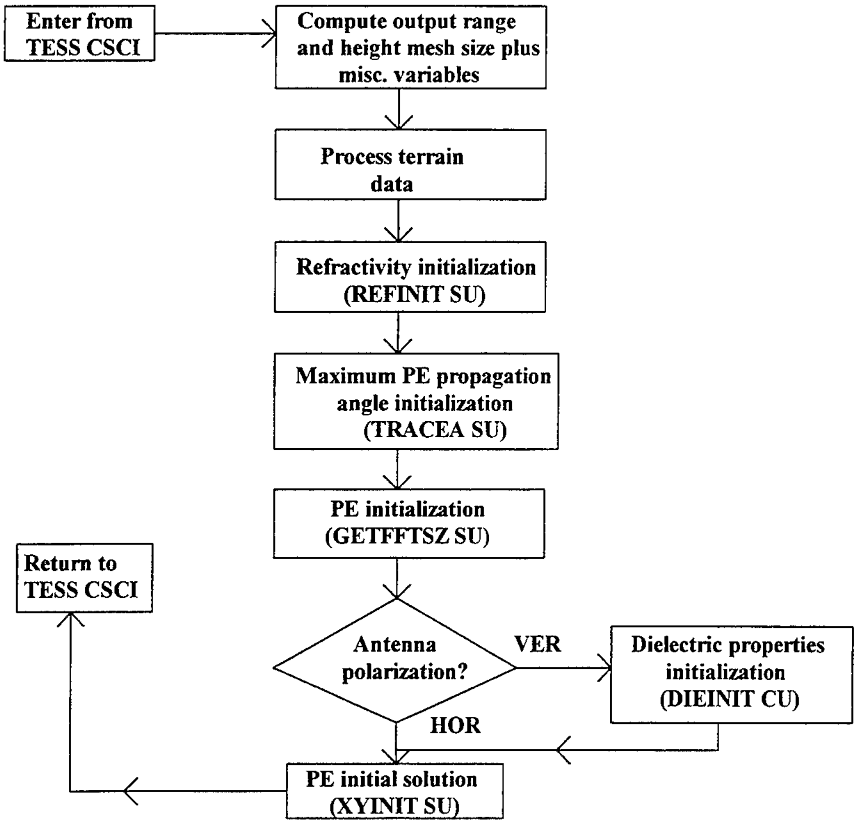

Figure 3-2 Program flow of the PEINIT CSC

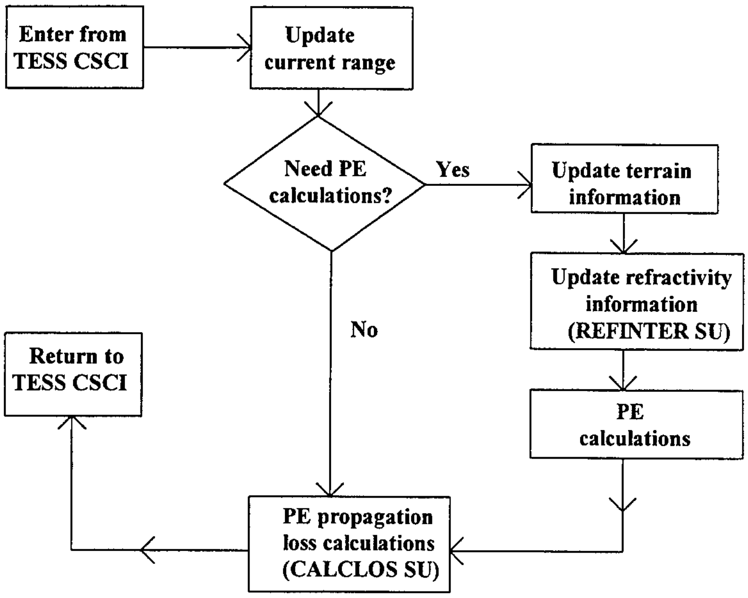

Figure 3-3 Program flow of the PESTEP CSC

The TPEM CSCI should be divided into 2 main computer software components (CSC) and 20 additional software units (SU). The name, purpose, and a general description of processing required for each SU follows.

3.1.1 Parabolic Equation Initialization (PEINIT) CSC

The purpose of the PEINIT CSC is to interface with various SUs for the complete initialization of the TPEM CSCI.

The atmospheric volume must be "covered" or resolved with a mesh of calculation points which will, as a matter of routine, exceed the height/range resolution requirements of the particular application of the TPEM CSCI. Upon entering the PEINIT CSC, a range and height mesh size per the TPEM CSCI output point is calculated from the number of TPEM outputs and the maximum CSCI range and height.

The terrain profile should be initially examined and unnecessary range/height points should be discarded if neighboring terrain slopes are redundant. The minimum terrain height should be determined, and then the entire terrain profile should be adjusted by this height so that all internal calculations are referenced to this height. This should be done in order to maximize the PE transform calculation volume.

A REFINIT SU should be referenced to initialize the TESS CSCI specified modified refractivity and also to test for valid environment profiles. A PROFREF SU should adjust the environment profiles by the internal reference height, and a INTPROF SU should define the modified refractivity at all PE vertical mesh points.

In order to automatically determine the maximum PE calculation angle, a TRACEA SU should be referenced. This should determine, via ray tracing, the minimum angle for which adequate coverage can be given with the specified terrain and environment profile. A GETFFTSZ SU should be referenced to determine the fast Fourier transform (FFT) size for the calculated angle and to initialize data elements within the PE region which are dependent on the size of the FFT. The minimum size for the FFT is determined from the Nyquist criterion.

A PE starting SU (XYINIT) and an antenna pattern factor SU (ANTPAT) should be referenced by the XYINIT SU to generate a first solution to the PE. A FFT SU should

be referenced for data elements required in obtaining the PE's starting solution. If vertical polarization is specified, then additional calculations should be performed in the starter solution using Kuttler and Dockery's mixed transform method (Ref. g). In this case a DIEINIT SU should be used to initialize dielectric ground constants. For general ground types, the permittivity and conductivity are calculated as a function of frequency from curve fits to the permittivity and conductivity graphs shown in recommendations and reports of the International Radio Consulting Committee (Ref. m).

A TRACEH SU should be referenced to determine the height at each output range step below which propagation loss solutions will be valid. No propagation loss solutions should be provided above these heights.

Finally, a PHASE1 SU should be referenced to initialize the free space propagator arrays, and a PHASE2 SU should be referenced (for range-independent environment profile) to initialize the environment propagator arrays.

3.1.1.1 Antenna Pattern (ANTPAT) SU

The purpose of an ANTPAT SU is to calculate a normalized antenna gain (antenna pattern factor) for a specified antenna elevation angle.

From the antenna beam width, elevation angle (an angle for which the antenna pattern factor is desired), and the antenna radiation pattern type; an antenna factor should be calculated.

3.1.1.2 Refractivity Initialization (REFINIT) SU

The purpose of the REFINIT SU is to check for valid environmental profile inputs and to initialize the refractivity arrays.

Upon entering, the environmental data should be checked for a range-dependent profile, and tested whether the range of the last profile entered is less than the maximum output range specified. If so, an error message should be returned, depending on the values of error flags set in the TESS CSCI itself.

It should also test for valid refractivity level entries for each profile. If the last gradient in any profile is negative, it should return an error message. If no errors are detected, the REFINIT SU then should extrapolate the environmental profiles vertically to 1000 km in height. Extrapolation should not be performed horizontally from the last provided profile; rather, the last provided environment profile should be duplicated at 10^7 km in range. This duplication of profiles is done by the REMDUP SU.

3.1.1.3 Trace for Minimum Angle (TRACEA) SU

The purpose of the TRACEA SU is to perform a ray trace to determine the minimum angle required (based on the reflected ray) in obtaining a PE solution for all heights up to the maximum output height (or the largest height allowed from the maximum transform size) and for all ranges beyond 90% of the maximum output range. The maximum PE propagation angle should then be determined from this angle. This should be done only for smooth surface and automatic angle calculation.

For terrain cases, the maximum PE propagation angle should have already been set to the larger of the critical angle (if a duct exists), the angle that clears the highest terrain peak, or the tangent angle determined from the maximum output height and the maximum output range.

If a maximum propagation angle is specified by the TESS CSCI, then the maximum PE propagation angle should be determined based on the given angle. However, a ray trace should still be performed in order to determine the initial launch angle such that the local angle of the ray remains less than the specified maximum propagation angle. The initial launch angle is to be used in the TRACEH SU.

3.1.1.4 Dielectric Initialization (DIEINIT) SU

The purpose of the DIEINIT SU is to determine the conductivity and relative permittivity as function of frequency in MHz based on general ground composition types.

3.1.1.5 Get FFT Size (GETFFTSZ) SU

The purpose of the GETFFTSZ SU is to determine the required transform size based on the maximum PE propagation angle and the specified maximum output height.

If the transform size required is greater than the maximum allowed, then the maximum PE height calculation volume should be calculated based on the maximum allowable transform size. Propagation loss should be provided only up to the maximum PE calculation height or the specified maximum output height, whichever is smaller.

For computational efficiency reasons, an artificial upper boundary must be established for the PE solution. To prevent upward propagating energy from being "reflected" downward from this boundary and contaminating the PE solution, the PE solution field strength should be attenuated or "filtered" above a certain height to insure that the field strength just below this boundary is reduced to zero.

Upon entering this SU, the total number of vertical points for which a transformation will be computed should be determined. This term is also referred to as the FFT size. The filtering boundary height should be determined.

3.1.1.6 Starter Field Initialization (XYINIT) SU

The purpose of the XYINIT SU is to calculate the complex PE solution at range zero.

Upon entering this SU, several constant terms which will be employed over the entire PE mesh should be calculated. These are the angle difference between mesh points in p-space; a height-gain value at the source (transmitter); and the complex index of refraction (if using vertical polarization). The complex index of refraction for vertical polarization is obtained from the GETALN SU.

For each point in the PE p-space mesh, the following steps should be performed:

(a) The antenna pattern ANTPAT SU should be referenced to obtain an antenna pattern factor for both a direct-path ray and a surface-reflected ray. The complex surface reflection coefficient should be determined with the direct-path ray elevation angle. Since the PE starting solution makes a flat-earth assumption, the direct-path ray elevation angle should be used in place of the surface-grazing angle.

(b) The complex portions of the PE solution should be determined from the antenna pattern factors, reflection coefficient, elevation angle, and gain.

3.1.1.7 Fast-Fourier Transform (FFT) SU

The purpose of the FFT SU is to separate the real and imaginary components of the complex PE field into two real arrays and then to reference the SINFFT SU which transforms each portion of the PE solution.

3.1.1.8 Sine Fast-Fourier Transform (SINFFT) SU

A function with a common period, such as a solution to the wave equation, may be represented by a series consisting of sines and cosines. This representation is known as a Fourier series. An analytical transformation of this function, known as a Fourier transform, may be used to obtain a solution for the function.

The solution to the PE approximation to Maxwell's wave equation is to be obtained by using such a Fourier transformation function. The TPEM CSCI requires only the real-valued sine transformation in which the real and imaginary parts of the PE equation are transformed separately. A Fourier transformation for possible use with the TPEM CSCI is described by Bergland (Ref. a) and Cooley, Lewis, and Welsh (Ref. b).

3.1.1.9 Trace Launch Angle (TRACEH) SU

The purpose of the TRACEH SU is to perform a ray trace for a single ray and store all heights traced to each output range step. The initial launch angle should be determined from the maximum PE propagation angle. The heights are stored for subsequent output of propagation loss values. This is done so that only loss values that fall within the valid PE solution region are output.

3.1.1.10 Free-Space Propagator Phase Term (PHASE1) SU

The purpose of the PHASE1 SU is to initialize the free-space propagator array for subsequent use in the PESTEP SU. The propagator term should be computed at each PE angle, or p-space, mesh point using the wide-angle propagator. Finally, a filter, or attenuation function (frequently called "window"), should be applied to the upper one-quarter of the array corresponding to the highest one-quarter of the maximum propagation angle.

3.1.1.11 Environmental Propagator Phase Term (PHASE2) SU

The purpose of the PHASE2 SU is to calculate the environmental phase term for an interpolated environment profile. This environmental phase term should be computed at each PE height, or z-space, mesh point. Finally, a filter, or attenuation function (frequently called "window"), should be applied to the upper one-quarter of the array corresponding to the highest one-quarter of the calculation height domain.

3.1.1.12 Profile Reference (PROFREF) SU

The purpose of the PROFREF SU is to adjust the current refractivity profile so that it is relative to a reference height. The reference height should be initially the minimum height of the terrain profile. Upon subsequent calls from the PESTEP SU, the refractivity profile should be adjusted by the local ground height at each PE range step.

3.1.1.13 Interpolate Profile (INTPROF) SU

The purpose of the INTPROF SU is to perform a linear interpolation vertically with height on the refractivity profile. Interpolation should be performed at each PE mesh height point.

3.1.2 Parabolic Equation Step (PESTEP) CSC

The purpose of the PESTEP CSC is to advance the entire TPEM CSCI algorithm one output range step, referencing various SUs to calculate the propagation loss at the current output range.

Upon entering the PESTEP CSC, the next output range should be determined and an iterative loop begun to advance the PE solution such that for the current PE range, a PE solution is calculated from the solution at the previous PE solution range. This procedure is to be repeated until the output range is reached.

At each PE range step, the local ground height should be determined and the PE field should be "shifted" by the number of bins, or PE mesh height points, corresponding to the local ground height. This is done in a DOSHIFT SU.

If using vertical polarization, a GETALN SU should be referenced to determine the impedance term and all associated variables used for the mixed transform calculations.

If the TPEM CSCI is being used in a range-dependent mode, that is, more than one profile has been input; or a terrain profile is specified, the REFINTER SU should be referenced to compute a new modified refractive index profile adjusted by the local ground height at the current range. The PHASE2 SU should be then referenced to compute a new environmental phase term using this new refractivity profile.

Using a FRSTP SU, the PE solution is transformed to p-space, advanced by the free space propagator array, and transformed back to z-space. The environmental phase term should be then applied to obtain the new final PE solution at the current range. Finally, a CALCLOS SU should be referenced to obtain the propagation loss at the desired output heights at the current output range.

3.1.2.1 DOSHIFT SU

The purpose of the DOSHIFT SU is to shift the field by the number of bins, or PE mesh heights corresponding to local ground height.

Upon entry, the number of bins to be shifted should be determined. The PE solution should then be shifted downward if the local ground is currently at a positive slope, and upward if the local ground is at a negative slope.

3.1.2.2 GETALN SU

The purpose of the GETALN SU is to compute the impedance term in the Leontovich boundary condition, and the complex index of refraction for finite conductivity and vertical polarization calculations. These formulas follow Kuttler and Dockery 's method (Ref. g).

3.1.2.3 Free Space Range Step (FRSTP) SU

The purpose of the FRSTP SU is to propagate the complex PE solution field in free space by one range step.

Upon entry the PE field should be transformed to p-space and then multiplied by the free space propagator. Before exiting the PE field should be transformed back to z-space. Both transforms are to be performed using a FFT SU.

3.1.2.4 Refractivity Interpolation (REFINTER) SU

The purpose of the REFINTER SU is to interpolate both horizontally and vertically on the modified refractivity profiles. Profiles are then adjusted so they are relative to the local ground height .

Upon entry, if there is a range-dependent environment, horizontal interpolation in range to the current PE range should be performed between the two neighboring TESS CSCI profiles that are specified relative to mean sea level. A REMDUP SU should be referenced to remove duplicate refractivity levels, and the PROFREF SU should then be referenced to adjust the new profile relative to the internal reference height corresponding to the minimum height of the terrain profile. The PROFREF SU should then be referenced once more to adjust the profile relative to the local ground height, and upon exit from the PROFREF SU, the INTPROF SU should be referenced to interpolate vertically on the refractivity profile at each PE mesh height point.

3.1.2.5 Remove Duplicate Refractivity Levels (REMDUP) SU

The purpose of the REMDUP SU is to remove any duplicate refractivity levels in the currently interpolated profile.

3.1.2.6 Calculate Propagation Loss (CALCLOS) SU

The purpose of the CALCLOS SU is to determine the propagation loss at each output height point at the current output range.

Upon entry the local ground height at the current output range is determined. All propagation loss values at output height points up to the local ground height should be then set to zero. The first valid loss point should be determined corresponding to the first output height point above the ground height. Next, the last valid loss point should be determined based on the smaller of the maximum output height or the height traced along the maximum PE propagation angle to the current output range.

From the height of the first valid loss point to the height of the last valid loss point, the GETPFAC SU should be referenced to obtain the propagation factor in dB (field strength relative to free space) at all corresponding output heights at the previous PE range and at the current PE range. Then, for each valid output height, horizontal interpolation in range should be performed to obtain the propagation factor at the current output range. From the propagation factor and the free-space loss, the propagation loss at each valid output height should then be determined, with the propagation loss set to (-1.) for all output height points above the last valid output height but less than the maximum output height. All loss values returned to the TESS CSCI at this point should be in centibels (10 cB = 1 dB)

3.1.2.7 Get Propagation Factor (GETPFAC) SU

The purpose of the GETPFAC SU is to determine the propagation factor at the specified height in dB.

Upon entering, a vertical interpolation with height on the PE solution field should be performed to obtain the magnitude of the field at the desired output height point. An additional calculation should be made and the propagation factor should then be returned in dB.

3.2 CSCI External Interface Requirements

The TPEM CSCI is accessed, through the PEINIT CSC, by a subroutine call from the TESS CSCI which should provide, as global data elements, the values specified in Table 3-1 through Table 3-4.

The TPEM CSCI external data elements, i.e. those data which must be provided by the calling TESS CSCI prior to the TPEM CSCI execution may be divided into four classifications. The first is external data related to the atmospheric environment, specified within Table 3-1; the second is data related to the EM system being assessed, specified within Table 3-2; the third is data related to the implementation of the TPEM CSCI by the TESS CSCI, specified within Table 3-3; and the fourth is data related to the terrain information, specified within Table 3-4. Each table lists the type, units, and bounds of each data element. Table 3-5 specifies the output data of the TPEM CSCI model.

Table 3-1 TPEM CSCI Environmental Data Element Requirements

Name	Description	Type	Units	Bounds
refmsl	Profile modified refractivity array referenced to mean sea level	real	M	≥ 0.0 [a]
hmsl	Profile height array	real	m	≥ 0.0 [a]
nprof	Number of profiles	integer	N/A	≥ 1
lvlep	Number of profile levels	integer	N/A	≥ 2
rngprof	Array of ranges to each profile	real	m	≥ 0.0

[a] Couplets of height and modified refractivity associated with that height are referred to within this document as an environmental profile.

Table 3-2 TPEM CSCI External EM System Data Element Requirements

Name	Description	Type	Units	Bounds
bwidth	Antenna vertical beam width	real	degree	0.5 to 45.0
elev	Antenna elevation angle	real	degree	-10.0 to 10.0
freq	EM system frequency	real	MHz	100.0 to 20,000.0
ipat	Antenna pattern 0 = Omni-directional 1 = Gaussian 2 = Sine (X)/X 3 = Cosecant-squared 4 = Generic height-finder	integer	N/A	0 to 4
polar	Antenna polarization H = Horizontal V = Vertical	character	N/A	'H' or 'V'
antht	Antenna height above local ground at range 0.0 m	real	m	≥ 1.0

Table 3-3 TPEM CSCI External Implementation Constants

Name	Description	Type	Units	Bounds
mxnfft	Maximum power of 2 for transform size	integer	N/A	≥ 9
maxpts	Maximum number points for FFT array	integer	N/A	$= 2^{mxnfft}$
mxlvls	Maximum number of profile levels for all possible applications of TPEM	integer	N/A	≥ 2 [b]
mxrout	Maximum number of range output points for all possible applications of TPEM	integer	N/A	≥ 1 [a]
nrout	Number of range output points for a particular application of TPEM	integer	N/A	= *mxrout* [a]
mxzout	Maximum number of height output points for all possible applications of TPEM	integer	N/A	≥ 1 [a]
nzout	Number of height output points for a particular application of TPEM	integer	N/A	= *mxzout* [a]
mxnprof	Maximum number of profiles for all possible applications of TPEM	integer	N/A	≥ 1 [b]
mxter	Maximum number of height/range points in terrain profile for all possible applications of TPEM	integer	N/A	≥ 2 [c]
lerr6	Logical flag to allow for error -6 to be bypassed	logical	N/A	'.true.' or '.false.' [a]
lerr12	Logical flag to allow for error -12 to be bypassed	logical	N/A	'.true.' or '.false.' [a]

Table 3-3: TPEM CSCI External Implementation Constants (cont'd)

Name	Description	Type	Units	Bounds
rmax	Maximum range output for a particular application of TPEM	real	m	≥ 5000.0 [b]
hmin	Minimum height output for a particular application of TPEM	real	m	≥ 0.0 [c]
hmax	Maximum height output for a particular application of TPEM	real	m	≥ 100.0 [b]
propang	Maximum PE propagation angle	real	deg	≥ 0.0 [a]

[a] refer to section 3.5.1 for a complete description.
[b] refer to section 3.5.2 for a complete description.
[c] refer to section 3.5.3 for a complete description.

Table 3-4 TPEM CSCI External Terrain Data Element Requirements

Name	Description	Type	Units	Bounds
terx	Terrain profile range array	real	m	≥ 0.0 [a]
tery	Terrain profile height array	real	m	≥ 0.0 [a]
itp	Number of terrain profile points for a particular application of TPEM	integer	N/A	$\leq mxter$ [a]
igr	Number of ground types for a particular application of TPEM	integer	N/A	≤ 50
igrnd	Array of ground composition types for a particular application of TPEM 0 = Sea water 1 = Fresh water 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = User defined	integer	N/A	$0 \leq igrnd \leq 5$ [a]
rgrnd	Array of ranges for which ground types are applied for a particular application of TPEM	real	m	≥ 0.0 [a]
dielec	2-dimensional array of relative permittivity and conductivity for a particular application of TPEM	real	N/A	>0 [a]
	[a] refer to Section 3.5.3 for a complete description.			

Table 3-5 TPEM CSCI Output Data Element Requirements

Name	Description	Type	Units	Source
hminter	Minimum elevation height of terrain profile	real	m	PEINIT CSC
ierror	Integer value that is returned if an error exists in input data	integer	N/A	PEINIT CSC
mloss	Propagation loss	integer	cB	PESTEP CSC
jstart	Output height index at which valid propagation loss values begin	integer	N/A	PESTEP CSC
jend	Output height index at which valid propagation loss values end	integer	N/A	PESTEP CSC
rout	Current range	real	m	PESTEP CSC
	[a]Refer to Section 3.5.1 for a complete description.			

3.3 CSCI Internal Interface Requirements

Section 3.1 shows the relationship between the TPEM CSCI and its two main CSCs PEINIT and PESTEP. This relationship is illustrated in Figure 3-1. The required internal interface between these two CSCs and the TPEM CSCI is left to the design. However, Table 5-1 should be used as a guide to the required internal interfaces in this CSCI.

3.4 CSCI Internal Data Requirements

The TPEM CSCI requires several internal arrays to be pre-dimensioned. Since the software documentation standards and coding requirements do not allow for dynamic dimensioning, these arrays must be dimensioned at program compile time. The implementation constants *mxrout* and *mxzout* refer to the extreme dimensions for various range and height related arrays for all possible applications of the TPEM CSCI and will be specified when the TPEM CSCI is compiled.

Due to the computational intensity of the TPEM CSCI, it may not be necessary or desirable to use the extreme capability of the TPEM CSCI for all applications. The variables *nrout* and *nzout* refer to the desired number of range and height output points for any one particular application, and will be specified when the PEINIT CSC is called.

For the TPEM CSCI implementation within the TESS coverage and loss diagram applications, *mxrout* and *nrout* should be set equal, and *mxzout* and *nzout* should be set equal. The values chosen for *mxrout* and *mxzout* must be applicable to the coverage diagram.

One of the parameters returned to the TESS application from the PEINIT CSC is *ierror.* This is to allow for greater flexibility in how input data is handled within the TESS application. Table 3-6 lists all possible errors that can be returned.

Table 3-6 PEINIT SU Returned Error Definitions

error	Definition
-6	Last range in terrain profile is less than *rmax.* Will only return this error if *lerr6* set to '.true.'
-8	*hmax* is less than maximum height of terrain profile
-12	Range of last environment profile given (for range-dependent case) is less than *rmax*. Will only return this error if *lerr12* set to '.true.'
-14	Last gradient in any environment profile is negative
-17	Range points of terrain profile are not increasing
-18	First range value in terrain profile is not 0.

The logical variables *lerr6* and *lerr12,* when set to '.false.', allow the TESS application to bypass their associated errors as these are not critical to the operation of the TPEM CSCI.

The TPEM CSCI provides propagation loss for moderately low angles and heights. It does not provide propagation loss for *all* heights and ranges desired. Propagation loss values will be provided automatically at all heights from *at least* 90% of the desired maximum range, *rmax,* to *rmax*. At lesser ranges, propagation loss values will be reduced in height. If greater coverage is desired (i.e., propagation loss desired at closer ranges and larger heights), a non-zero positive value for *propang* must be specified. The TPEM CSCI will then provide propagation loss for all heights and ranges up to the maximum angle specified by *propang.* Set *propang* equal to zero to use the TPEM CSCI's default angle and coverage.

3.5 Adaptation Requirements

3.5.1 Environmental Radio Refractivity Field Data Elements

The radio-refractivity field, i.e. the profiles of M-units versus height, should consist of vertical piece-wise linear profiles specified by couplets of height in meters

above mean sea level and modified refractivity (M-units) at multiple arbitrary ranges. All vertical profiles must contain the same number of vertical data points, and be specified such that each numbered data point corresponds to like-numbered points (i.e. features) in the other profiles. The first numbered data point of each profile must correspond to a height of zero mean sea level and the last numbered data point must correspond to a height such that the modified refractivity for all greater heights is well represented by extrapolation using the two highest profile points specified. Within each profile, each numbered data point must correspond to a height greater than or equal to the height of the previous data point. Note that this requirement allows for a profile which contains redundant data points. Note also that all significant features of the refractivity profiles must be specified, even if they are above the maximum output height specified for a particular application of TPEM.

The TESS CSCI application designer and the TESS operator share responsibility for determining appropriate environmental inputs. For example, a loss diagram may be used to consider a surface-to-surface radar detection problem. Since the operator is interested in surface-to-surface, he may truncate the profile assuming that effects from elevated ducting conditions are negligible. It may be however, that the elevated duct does indeed produce a significant effect. The operator should insure therefore, that the maximum height of the profile allows for the inclusion of all significant refractive features.

This specification allows a complicated refractivity field to be described with a minimum of data points. For example, a field in which a single trapping layer linearly descends with increasing range can be described with just two profiles containing only four data points each, frame (a) of Figure 3-4. In the same manner, other evolutions of refractive layers may be described. Frames (b) and (c) of Figure 3-4 show two possible scenarios for the development of a trapping layer. The scenario of choice is the one which is consistent with the true thermodynamical and hydrological layering of the atmosphere.

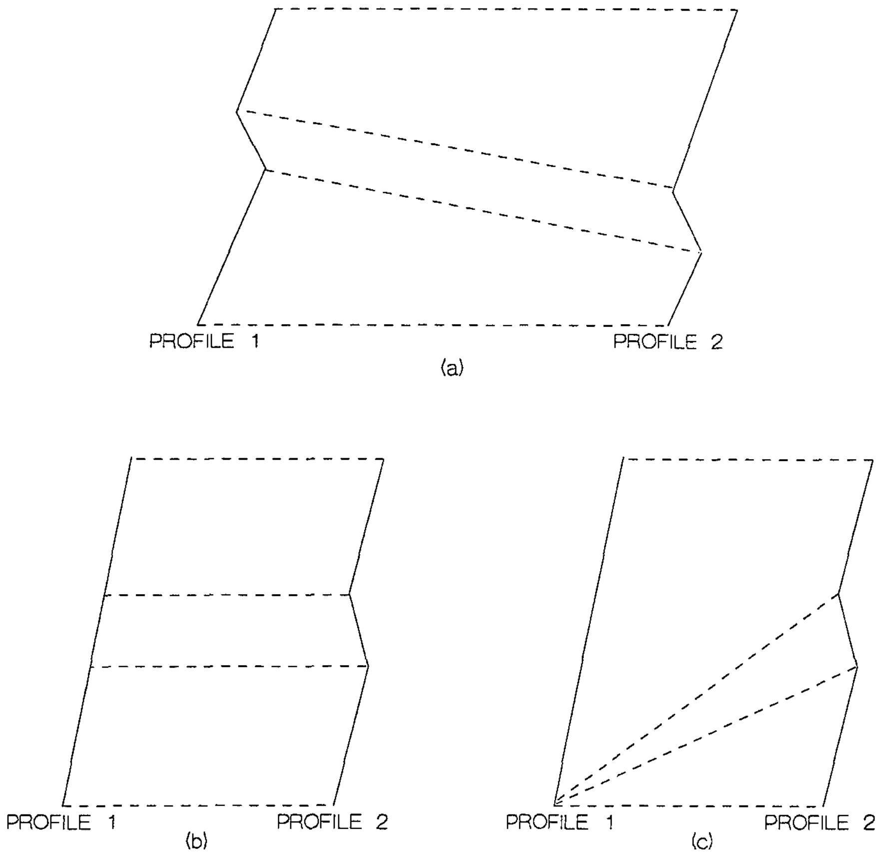

Figure 3-4. Idealized M-unit profiles (solid) and lines of interpolation (dashed)

The two TPEM CSCI implementation constants *mxlvls* and *mxnprof* refer to the maximum number of height levels allowed within a profile and the maximum number of profiles allowed by the TPEM CSCI. These two constants must be specified when the TPEM CSCI is compiled and be carefully chosen to be just large enough for all calling applications but small enough to efficiently conserve both computer memory and execution time of the TPEM CSCI. While there is no upper limit on *mxlvls* imposed by the TPEM CSCI, increasing the number of environmental levels will increase the TPEM CSCI execution time. Increasing the execution time for any particular application denies

valuable computer resources to other applications, and thus makes the application less likely to be used by an operator.

Two external implementation data variables applicable to both the TESS operator and to the calling application designer are *rmax*, the maximum TPEM CSCI output range, and *hmax*, the maximum TPEM CSCI output height. These two parameters are required by the TPEM CSCI to determine the horizontal and vertical resolution, respectively, for internal range and height calculations based on the current values of *nrout* and *nzout*. Any value of *rmax* and *hmax* is allowed for the convenience of the TESS operator and the calling application designer. For example, the TESS operator may desire a coverage diagram which extends to a range of 500 kilometers (km). In addition to accommodating the desires of the operator, specification of such a convenient maximum range eases the burden for the application designer in determining incremental tick marks for the horizontal axis of the display.

Provided the value of the parameter *lerr12* is set to '.false.', if the furthest environment profile range is less than *rmax,* the TPEM CSCI will automatically create an environment profile at *rmax* equal to the last profile specified, making the environment homogeneous from the range of the last profile specified to *rmax*. For example, a profile is input with an accompanying range of 450 km. If the TESS operator chooses an *rmax* of 500 km, the TPEM CSCI will continue loss calculations to 500 km, keeping the refractivity environment homogeneous from 450 km to 500 km.

If *lerr12* is set to '.true.' and the furthest environment profile range is less than *rmax*, then an error will be returned in *ierror* from the PEINIT CSC. This is to allow the TESS CSCI application designer greater flexibility in how environment data is handled.

3.5.2 Terrain Profile Data Element

The terrain profile should consist of linear piece-wise segments specified in terms of range/height pairs. The TPEM CSCI implementation constant *mxter* refers to the maximum number of height/range pairs allowed within a terrain profile. This constant must be specified when the TPEM CSCI is compiled and must be carefully chosen to be just large enough for all calling applications but small enough to efficiently conserve both computer memory and execution time of the TPEM CSCI. While there is no upper limit

on *mxter* imposed by the TPEM CSCI, increasing the number of terrain points will increase the TPEM CSCI execution time. Increasing the execution time for any particular application denies valuable computer resources to other applications, and thus makes the application less likely to be used by an operator.

All range values must be increasing, and the first terrain height value must be at range zero. General ground composition types can be specified (Table 3-4), along with corresponding ranges over which the ground type is to be applied. If ground type "User Defined" is specified (*igrnd()* = 5), then numeric values of relative permittivity and conductivity must be given. If horizontal antenna polarization is specified, the TPEM CSCI will assume perfect conductivity for the entire terrain profile and will ignore any information regarding ground composition. If vertical antenna polarization is specified, then information regarding ground composition must also be specified.

The maximum height, *hmax*, must always be greater than the minimum height, *hmin*. Also, a value of *hmax* must be given such that it is larger than the maximum elevation height along a specified terrain profile.

Provided *lerr6* is set to '.false.', if the furthest range point in the terrain profile is less than *rmax*; the TPEM CSCI should automatically create a height/range pair as part of the terrain profile at *rmax* with elevation height equal to the last height specified in the profile, making the terrain profile flat from the range of the last profile point specified to *rmax*. For example, a terrain profile is input where the last height/range pair is 50 meters (m) in height with an accompanying range of 95 km. If the TESS operator chooses an *rmax* of 100 km, the TPEM CSCI should continue loss calculations to 100 km, keeping the terrain profile flat from 95 km to 100 km with an elevation height of 50 m.

If *lerr6* is set to '.true.' and the furthest range point is less than *rmax*, then an error should be returned in *error* from the PEINIT SU. This is to allow the TESS CSCI application designer greater flexibility in how terrain data is handled.

3.6 Security and Privacy Requirements

The security and privacy requirements are the same as those required by the target employing TESS CSCI.

3.7 CSCI Environmental Requirements

The TPEM CSCI must be able to operate in the same hardware and software environments that the target employing TESS CSCI operates.

3.8 Computer Resource Requirements

Section 3.1.1.8 describes a requirements for a Sine Fast-Fourier Transform (SinFFT) SU. However, other sine FFT routines are available in the commercial market, and such a sine FFT may already be available within another TESS CSCI. The selection of which FFT ultimately used by TPEM CSCI is left to the application designer as every sine FFT will have hardware and/or software performance impacts.

3.9 Software Quality Factors

The primary required quality factors can be divided into the three categories - design, performance, and adaptation.

The quality factors for the design category should include: correctness, maintainability, and verifiability. Correctness describes the extent to which the TPEM CSCI conforms to its requirements and is to be determined from the criteria - completeness, consistency, and/or traceability. Maintainability specifies the effort required to locate and fix an error in the TPEM CSCI. Maintainability is to be determined from the criteria - consistency, modularity, self-descriptiveness (self-documentation), and/or simplicity. Verifiability characterizes the effort required to test the TPEM CSCI to ensure that it performs its intended function. Verifiability is to be determined from the criteria - modularity, self-descriptiveness, and/or simplicity.

The quality factor for performance category is reliability, which depicts the confidence that can be placed in the TPEM CSCI calculations. Reliability is to be

determined from the criteria - accuracy, anomaly management, auditability, consistency, and/or simplicity.

The quality factors for the adaptation category are portability and reusability. Portability determines how easy it is to transport the TPEM CSCI from one hardware and/or software environment to another. Portability is to be determined from the criteria - application independence, modularity, and/or self-descriptiveness. Reusability illustrates how easy it is to convert the TPEM CSCI (or parts of the CSCI) for use in another application. Reusability is to be determined from the criteria - application independence, document accessibility, functional scope, generality, hardware independence, modularity, simplicity, self-descriptiveness, and/or system clarity.

Section 7.1 of APPENDIX A defines the software quality criteria.

Only the software quality criteria completeness, consistency, and traceability can be analyzed. Their calculation is described in Section 7.2 of APPENDIX A. The other criteria have to be determined by either demonstration, test, or inspection.

3.10 Design And Implementation Constraints

3.10.1 Implementation And Application Considerations

The calling TESS CSCI application will determine the employment of the TPEM CSCI. However, the intensive computational nature of the TPEM CSCI must be taken into consideration when designing an efficient calling application. For this reason, the TPEM CSCI should be designed with flexibility for various hardware suites and computer resource management considerations. As stated in Section 1.1 on page 1, this TPEM CSCI applies only to a coverage and loss diagram application. The following highly recommended guidelines are provided to aid in the design of a coverage or loss diagram application which will most efficiently employ the TPEM CSCI.

The TPEM CSCI propagation loss calculations are independent of any target or receiver considerations, therefore, for any EM emitter, one execution of the TPEM CSCI may be used to create both a coverage diagram and a loss diagram. Since both execution time and computer memory allocation should be a consideration when employing this

model, it is most efficient and appropriate to execute the TPEM CSCI for a particular EM system/environmental/terrain combination before executing any application. The output of the TPEM CSCI would be stored in a file which would be accessed by multiple applications.

For example, the TESS operator may desire a coverage diagram for one particular radar system. At the beginning of the coverage diagram application, a check would be made for the existence of a previously created TPEM CSCI output file appropriate for the EM system, environmental, and terrain conditions. If such a file exists, the propagation loss values would be read from the file and used to create the coverage diagram. If the file does not exist, the TPEM CSCI would be executed to create one. As the TPEM CSCI is executing, its output could be routed simultaneously to a graphics display device and a file. This file could then be used in the loss diagram application should the operator also choose it. Two distinct applications therefore, are achieved with only one execution of the TPEM CSCI. Additionally, should the operator desire an individual coverage diagram for each of multiple targets, or a single coverage diagram illustrating radar detection of a low-flying missile superimposed upon a coverage diagram illustrating his own radar's vulnerability as defined by the missile's ESM receiver, only a single execution of the TPEM CSCI would be required, thereby saving valuable computer resources.

3.10.2 Programming Language And Source Implementation

3.10.2.1 Programming Language

The ANSI Fortran 90 program language standard must be used in the development of the TPEM CSCI (Ref h). This standard consists of the specifications of the language Fortran. With certain limitations the syntax and semantics of the old International Standard commonly known as "FORTRAN 77" are contained entirely within this new International Standard. Therefore, any standard-conforming FORTRAN 77 program is standard conforming under the Fortran 90 Standard. Note that the name of this language, Fortran, differs from that in FORTRAN 77 in that only the first letter is capitalized. The **Overview** section of the International Standard describes the major additions to FORTRAN 77 in this International Standard. Section 1.3 of the International Standard specifies the bounds of the Fortran language by identifying both those items included and

those items excluded. Section 1.4.1 describes the FORTRAN 77 compatibility of the International Standard with emphasis on four FORTRAN 77 features having different interpolations in the new International Standard. The International Standard provides facilities that encourage the design and the use of modular and reusable software.

Section 8.2 of the International Standard describes nine obsolescent features of FORTRAN 77 that are redundant and for which better methods are available in FORTRAN 77 itself. These nine obsolescent features should not be used, These obsolescent features are:

(a) **Arithmetic IF** - use the **IF** statement.

(b) Real and double precision **DO** control variables and **DO** loop control expressions - use integer.

(c) Shared **DO** termination and termination on a statement other than **END DO** or **CONTINUE** - use an **END DO** or a **CONTINUE** statement for each **DO** statement.

(d) Branching to an **END IF** statement from outside its IF block - branch to the statement following the **END IF**.

(e) Alternate return.

(f) **PAUSE** statement.

(g) **ASSIGN** and assigned **GO TO** statements.

(h) Assigned **FORMAT** specifiers.

(i) cH (nH) edit descriptor.

Remedies for the last five obsolescent features are described in section 8.2 of the International standard.

3.10.2.2 Source Implementation

Ref. (f) by the Naval Oceanographic Office establishes a uniform standard for all software submitted by all contributors to them. It is recommended that the coding requirements set forth in Section 4 of that document be followed. Among these recommendations are:

(a) Special non-ANSI features shall be avoided. Non-ANSI practices that are necessary must be documented in the code itself.

(b) Maximum use should be made of existing commercially available FORTRAN callable libraries.

(c) Programs shall be designed and coding using only five basic control structures - sequence of operations (assignment, add, ...), **IF THEN ELSE**, **DO WHILE**, **DO UNTIL**, and **CASE**.

(d) Procedures or routines that make up a module shall not exceed an average of 100 executable statements per procedure or routine and shall not exceed a maximum of 200 executable statements in any procedure or routine.

(e) Branching statements (**GO TO**s) shall only pass control to a statement that is in the same procedure or routine. Each **GO TO** must pass control only forward of its point of occurrence.

(f) Naming conventions shall be uniform throughout the software. Program, subprogram, module, procedure, and data names shall be uniquely chosen to identify the applicable function performed. The naming convention for **COMMON** shall be consistent across the entire program.

(g) Constants shall be defined not calculated (e.g., do no use *HALF = 1/2*, use *HALF = 0.5*)

(h) Mixed-mode numerical operations should be avoided whenever possible. When determined to be necessary, the use shall be explicit (*FLOAT*, *FIX*, or in assignment statement) and completely described in comments.

(i) Each component of the software shall have a prologue containing the name of the program, subprogram, or function and any version number; purpose; inputs; outputs; list of routines that call this routine; complete list of routines called including intrinsic functions such as *ABS* and *FLOAT*; glossary; and method.

(j) To facilitate program comprehension, comment statements shall be used throughout the program code.

(k) The use of the **EQUIVALENCE** statement shall be restricted to those where it either improves the readability of the code or the efficiency of the program. If the **EQUIVALENCE** statement is used, it must be fully documented in the prologue and inline comment statements.

(l) No machine-dependent techniques are allowed, unless there is no other way of doing the job.

(m) Initialize every variable before use.

(n) Do not depend on the values of "local" variables computed on a previous call to a routine.

(o) Program structural indentation shall be used to improve readability and clarity.

3.11 Personnel-Related Requirements

Not applicable.

3.12 Training Related Requirements

The employing target software personnel implementing this CSCI into the TESS CSCI will require training to become familiar with the TPEM. This requirement should be met by this document and the companion Software Design Description (SDD) and Software Test Description (STD) documents.

3.13 Other Requirements

None.

3.14 Precedence And Criticality Of Requirements

The requirements presented in Sections 3.1 through 3.5 and Sections 3.8 through 3.10 have precedence over Sections 3.6, 3.7, 3.11, 3.12, and 3.13 and should be given equal weight.

4. QUALIFICATION PROVISIONS

N/A

5. REQUIREMENTS TRACEABILITY

5.1 System Traceability

This section provides traceability of requirements between the TPEM CSCI and the TESS CSCI.

(a) The TPEM CSCI environmental data requirements should be obtained from the Tactical Environmental Data System database (TEDS) within the TESS CSCI. The TPEM CSCI terrain data element requirements should be obtained from the Digital Terrain Elevation Database (DTED) within the TESS CSCI. The radar/communication system data element requirements should be obtained from the EM system database within the TESS CSCI.

(b) The TESS CSCI requirement of propagation loss vs. range and height should be obtained from the TPEM CSCI.

5.2 Documentation Traceability

This section provides the following types of traceability between the Software Requirements Specification (SRS), the Software Design Description (SDD), and the Software Test Description (STD):

(a) Traceability between levels of requirements;

(b) Traceability between the software requirements and software design;

(c) Traceability between the software requirements and qualification test information obtained from the software testing.

This traceability is presented in three tables. The first table, Table 5-1, presents the traceability between levels of SRS requirements. The second table presents the traceability between the software requirements and software design. The third table presents the traceability between the SRS requirements and the software test information.

Table 5-1 Requirements Traceability Matrix for the SRS

Software Requirements Specification		**Software Requirements Specification**	
SRS Requirement Name	SRS Paragraph Number	SRS Requirement Name	SRS Paragraph Number
CSCI Capability Requirements	3.1	Parabolic Equation Initialization (PEINIT) CSC	3.1.1
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Refractivity Initialization (REFINIT) SU	3.1.1.2
Refractivity Initialization (REFINIT) SU	3.1.1.2	Remove Duplicate Refractivity Levels (REMDUP) SU	3.1.2.5
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Trace for Minimum Angle (TRACEA) SU	3.1.1.3
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Dielectric Initialization (DIEINIT) SU	3.1.1.4
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Get FTT Size (GETFFTSZ) SU	3.1.1.5
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Starter Field Initialization (XYINIT) SU	3.1.1.6
Starter Field Initialization (XYINIT) SU	3.1.1.6	Antenna Pattern (ANTPAT) SU	3.1.1.1
Starter Field Initialization (XYINIT) SU	3.1.1.6	GETALN SU	3.1.2.2
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Fast-Fourier Transform (FFT) SU	3.1.1.7
Fast-Fourier Transform (FFT) SU	3.1.1.7	Sine Fast-Fourier Transform (SINFFT) SU	3.1.1.8
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Trace Launch Angle (TRACEH) SU	3.1.1.9
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Free-Space Propagator Phase Term (PHASE1) SU	3.1.1.10

Table 5-1 Requirements Traceability Matrix for the SRS (cont'd)

Software Requirements Specification		**Software Requirements Specification**	
SRS Requirement Name	SRS Paragraph Number	SRS Requirement Name	SRS Paragraph Number
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Environmental Propagator Phase Term (PHASE2) SU	3.1.1.11
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Profile Reference (PROFREF) SU	3.1.1.12
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Interpolate Profile (INTPROF) SU	3.1.1.13
CSCI Capability Requirements	3.1	Parabolic Equation Step (PESTEP) SU	3.1.2
Parabolic Equation Step (PESTEP) CSC	3.1.2	DOSHIFT SU	3.1.2.1
Parabolic Equation Step (PESTEP) CSC	3.1.2	GETALN SU	3.1.2.2
Parabolic Equation Step (PESTEP) CSC	3.1.2	Free Space Range Step (FRSTP) SU	3.1.2.3
Free Space Range Step (FRSTP) CSC	3.1.2.3	Fast-Fourier Transform (FFT) SU	3.1.1.7
Parabolic Equation Step (PESTEP) CSC	3.1.2	Refractivity Interpolation (REFINTER) SU	3.1.2.4
Refractivity Interpolation (REFINTER) SU	3.1.2.4	Profile Reference (PROFREF) SU	3.1.1.12
Refractivity Interpolation (REFINTER) SU	3.1.2.4	Interpolate Profile (INTPROF) SU	3.1.1.13
Refractivity Interpolation (REFINTER) SU	3.1.2.4	Remove Duplicate Refractivity Levels (REMDUP) SU	3.1.2.5
Parabolic Equation Step (PESTEP) CSC	3.1.2	Calculate Propagation Loss (CALCLOS) SU	3.1.2.6
Calculate Propagation Loss (CALCLOS) SU	3.1.2.6	Get Propagation Factor (GETFAC) SU	3.1.2.7
Parabolic Equation Step (PESTEP) CSC	3.1.2	Fast-Fourier Transform (FFT) SU	3.1.1.7
Parabolic Equation Step (PESTEP) CSC	3.1.2	Environmental Propagator Phase Term (PHASE2) SU	3.1.1.11

Table 5-1 Requirements Traceability Matrix for the SRS (cont'd)

Software Requirements Specification		**Software Requirements Specification**	
SRS Requirement Name	SRS Paragraph Number	SRS Requirement Name	SRS Paragraph Number
CSCI Capability Requirements	3.1	CSCI External Interface Requirements	3.2
CSCI Capability Requirements	3.1	CSCI Internal Interface Requirements	3.3
CSCI Capability Requirements	3.1	CSCI Internal Data Requirements	3.4
CSCI Capability Requirements	3.1	Adaptation Requirements	3.5
CSCI Capability Requirements	3.1	Security and Privacy Requirements	3.6
CSCI Capability Requirements	3.1	CSCI Environmental Requirements	3.7
CSCI Capability Requirements	3.1	Computer Resource Requirements	3.8
CSCI Capability Requirements	3.1	Software Quality Factors	3.9
CSCI Capability Requirements	3.1	Design And Implementation Constraints	3.10
Design And Implementation Constraints	3.10	Implementation and Application Considerations	3.10.1
Design And Implementation Constraints	3.10	Programming Language And Source Code Implementation	3.10.2
Programming Language And Source Code Implementation	3.10.2	Programming Language	3.10.2.1
Programming Language And Source Code Implementation	3.10.2	Source Implementation	3.10.2.2
CSCI Capability Requirements	3.1	Personnel-Related Requirements	3.11

Table 5-1 Requirements Traceability Matrix for the SRS (cont'd)

Software Requirements Specification		**Software Requirements Specification**	
SRS Requirement Name	SRS Paragraph Number	SRS Requirement Name	SRS Paragraph Number
CSCI Capability Requirements	3.1	Training Related Requirements	3.12
CSCI Capability Requirements	3.1	Other Requirements	3.13
CSCI Capability Requirements	3.1	Precedence and Criticality of Requirements	3.14

6. NOTES

Table 6-1 is a glossary of acronyms and abbreviations used within this document. Table 6-2 is a glossary of Fortran terms used within this document.

Table 6-1 Acronyms and Abbreviations

Term	Definition
ANSI	American National Standards Institute
cB	centibel
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
dB	decibel
EM	Electromagnetic
FFT	Fast-Fourier Transform
Fortran	Formula Translation
FORTRAN	Formula Translation
km	kilometers
m	meters
M	modified refractivity units
MHz	megahertz
N/A	not applicable
PE	Parabolic Equation
p-space	phase space
rad	radians
SDD	Software Design Description
SRS	Software Requirements Specification
STD	Software Test Description
SU	Software Unit
TESS	Tactical Environmental Support System
TPEM	Terrain Parabolic Equation Model
z-space	distance space

Table 6-2 Fortran Terms

Term	Action or Definitions
ABS	Absolute value function
Arithmetic IF	Transfers control to one of three statement labels, depending on the value of *expression*
ASSIGN	Assigns the value of a format or statement label to an integer variable
CASE	Marks the beginning of a block of statements executed if an item in a list of expressions matches the test expressions
COMMON	Allows two or more program units to directly share variables without having to pass them as arguments
CONTINUE	Does not have any effect
DO	Repeatedly executes the statements following the **DO** statement through the statement which marks the end of the loop
DO WHILE	Executes a block of statements repeatedly while a logical condition remains true
END DO	Terminates a **DO** or **DO WHILE** loop
END IF	Terminates a block of **IF** statements
EQUIVALENCE	Causes two or more variables or arrays to occupy the same memory location
FIX	Data type conversion function
FLOAT	Data type conversion function
FORMAT	Sets the format in which data is written to or read from a file
GO TO	Transfers execution to the statement label assigned to variable
IF	If expression is true, statement is executed; if expression is false, program execution continues with the next executable statement
IF THEN ELSE	If expression is true, statements in the **IF** block are executed; if expression is false, control is transferred to the next **ELSE**, **ELSE IF**, or **END IF** statement at the same IF level
PAUSE	Temporarily suspends program execution and allows you to execute operating system commands during the suspension

7. APPENDIX A

7.1 Definitions of Quality Factor Criteria

The criteria for judging the quality factors of Section 3.9 have the following definitions:

(a) Accuracy. The precision of computations and control;

(b) Anomaly management. The degree to which the program detects failure in order to maintain consistency;

(c) Application independence. The degree to which the program is independent of nonstandard programming language features, operating system characteristics, and other environmental constraints;

(d) Auditability. The ease with which conformance to standards can be checked;

(e) Completeness. The degree to which full implementation of required function has been achieved;

(f) Consistency. The use of uniform design and documentation techniques throughout the software development project;

(g) Document accessibility. The availability of documents describing the program components.

(h) Functional scope. The generality of the feature set and capabilities of the program;

(i) Generality. The breadth of potential application of program components;

(j) Hardware independence. The degree to which the software is decoupled from the hardware on which it operates;

(k) Modularity. The functional independence of program components;

(l) Self- descriptiveness. The degree to which the source code provides meaningful documentation;

(m) Simplicity. The degree to which a program can be understood without difficulty;

(n) System clarity. The ease for which the feature set and capabilities of the system can be determined.

(o) Traceability. The ability to trace a design representation or actual program component back to requirements.

7.2 Software Quality Metrics

7.2.1 Completeness Criteria

The criteria completeness can be determined from the metric:

(a) no ambiguous references (input, function, output);

(b) all data references defined;

(c) all referenced functions defined;

(d) all defined functions used;

(e) all conditions and processing defined for each decision point;

(f) all defined and referenced calling sequences parameters agree;

(g) all problem reports resolved;

(h) design agrees with requirements;

(i) code agrees with design;

(j) (score 0 for any untrue statement; 1 otherwise); and

(k) metric value = SUM (scores)/9.

7.2.2 Consistency Criteria

The criteria consistency can be determined from the metric : number of modules violating the design standard divided by the number of modules.

7.2.3 Traceability Criteria

The criteria traceability can be determined from the metric : number of itemized requirements traced divided by the total number of requirements.

SOFTWARE DESIGN DESCRIPTION

FOR THE

TERRAIN PARABOLIC EQUATION MODEL CSCI

May 1, 1997

Prepared for:

Space and Naval Warfare Systems Command (PMW-185)
Washington, DC

and

Naval Sea Systems Command (PEO USW ASTO-E/F)
Washington, DC

Prepared by:

Naval Command, Control and Ocean Surveillance Center
Research, Development, Test and Evaluation Division
Tropospheric Branch (Code D883)
San Diego, CA 92152-738

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. SCOPE

1.1 Identification

Terrain Parabolic Equation Model (TPEM) computer software configuration item (CSCI). The purpose of the TPEM CSCI is to calculate range-dependent electromagnetic (EM) system propagation loss within a heterogeneous atmospheric medium over variable terrain, where the radio-frequency index of refraction is allowed to vary both vertically and horizontally, also accounting for terrain effects along the path of propagation. Numerous Tactical Environmental Support System (TESS) applications require EM-system propagation loss values. The required TPEM model described by this document may be applied to two such TESS applications, one which displays propagation loss on a range versus height scale (commonly referred to as a coverage diagram) and one which displays propagation loss on a propagation loss versus range/height scale (commonly referred to as a loss diagram).

1.2 Document Overview

This document describes the design of the TPEM CSCI. An overview of the input software requirements is presented together with an overview of the CSCI design architecture and a detailed design description of each component of the CSCI.

2. REFERENCED DOCUMENTS

(a) Bergland, G. D., "A Radix-eight Fast Fourier Transform Subroutine for Real-valued Series," IEEE Trans. Audio and Electro-acoust., Vol. AU-17, pp. 138-144, 1969.

(b) Cooley, J. W., P. A. W. Lewis and P. D. Welsh, "The Fast Fourier Transform Algorithm: Programming Considerations in the Calculation of Sine, Cosine and Laplace Transforms," J. Sound Vib., Vol. 12, pp. 315-337, 1970.

(c) Tappert, F. D., "The Parabolic Approximation Method," Wave Propagation and Underwater Acoustics, J. B. Keller and J. S. Papadakis, Eds., New York, Springer-Verlag, pp. 224-285, 1977.

(d) CCIR, XVIth Plenary Assembly, Dubrovniks, "Electrical Characteristics of the Surface of the Earth; Recommendation 527-1," Recommendations and Reports of the CCIR, Vol. 5, Geneva, Int. Telecommun. Union, 1986

(e) Commander-In-Chief, Pacific Fleet Meteorological Requirement (PAC MET) 87-04, "Range Dependent Electromagnetic Propagation Models," 1987.

(f) Dockery, G. D., "Modeling Electromagnetic Wave Propagation in the Troposphere Using the Parabolic Equation," IEEE Trans. Antennas Propagat., Vol. 36, pp. 1464-14170, October 1988

(g) Naval Oceanographic Office, "Software Documentation Standards and Coding Requirements for Environmental System Product Development," April 1990.

(h) Kuttler, J. R. and G. D. Dockery., "Theoretical Description of the Parabolic Approximation/Fourier Split-Step Method of Representing Electromagnetic Propagation in the Troposphere," Radio Sci., Vol. 26, pp. 381-393, March-April, 1991.

(i) American National Standards Institute (ANSI), "Program Language - Fortran - Extended," 1992.

(j) Barrios, A. E., "Parabolic Equation Modeling in Horizontally Inhomogeneous Environments," IEEE Trans. Ant. and Prop., Vol. 40, pp. 791-797, July 1992.

(k) Barrios, A. E., "Terrain and Refractivity Effects on Non-Optical Paths," AGARD Conference Proceedings 543, Multiple Mechanism Propagation Paths (MMPPs): Their Characteristics and Influence on System Design, pp. 10- to 10-9, October 1993.

(l) Barrios, A. E., "A Terrain Parabolic Equation Model for Propagation in the Troposphere," IEEE Trans. Ant. and Prop., Vol. 42, pp. 90-98, January 1994.

(m) Naval Oceanographic Office, "Software Documentation Standards for Environmental System Product Development," Naval Oceanographic Office, Stennis Space Center, MS, OAML-SDS-59, February 1996.

(n) Barrios, A. E., "Terrain Parabolic Equation Model (TPEM) Version 1.5 User's Manual," Naval Command, Control and Ocean Surveillance Center, RDT&E Division, San Diego, CA, NRaD TD 2898, February 1996.

(o) Madere, S., Avery, D., Depner, J., Harrison, T., Hollingsworth, W., Kyle, M., Malley, D., Miller, G., Moseley, J., Ordish, D., Patman, T., and Robinson, J.,"Naval Oceanographic Office Programming Practices, Standards, and Conventions (PPS&C)," Naval Oceanographic Office, Stennis Space Center, MS, Report RP 56, June 1996.

3. CSCI-WIDE DESIGN DECISIONS

The designed TPEM CSCI propagation model is a pure split-step parabolic equation (PE) model that allows for range-dependent refractivity and variable terrain along the path of propagation. It calculates propagation loss both in range and altitude.

The TPEM CSCI provides propagation loss for moderately low angles and heights. It is not required to provide propagation loss for *all* heights and ranges desired. Propagation loss values can be provided at all heights from *at least* 90% of the desired maximum range to the maximum range. The TPEM CSCI allows for horizontal and vertical antenna polarizations, finite conductivity based on user-specified ground composition and dielectric parameters, and the complete range of EM system parameters and most antenna patterns required by TESS.

The TPEM CSCI is divided into 2 main computer software components (CSC) and 20 additional software units (SU). The purpose of the first CSC, the PEINIT CSC, is to interface with various SUs for the complete initialization of the TPEM CSCI. The purpose of the second CSC, the PESTEP CSC, is to advance the entire TPEM CSCI algorithm one output range step, referencing various SUs to calculate the propagation loss at the current output range.

4. CSCI ARCHITECTURE DESIGN

4.1 CSCI Components

The TPEM CSCI is accessed by a subroutine call which provides, as global data elements, the values specified in Tables 4-1 through 4-4.

The TPEM CSCI is divided into 2 computer software components (CSC) and 20 software units (SU). The two CSCs are the PEINIT CSC and the PESTEP CSC. The source code for the TPEM CSCI is listed in APPENDIX A. The name and purpose for each CSC and SU follows.

Parabolic Equation Initialization (PEINIT) CSC - to interface with various SUs for the complete initialization of the TPEM CSCI. The PEINIT CSC component SUs include:

(1) Antenna Pattern (ANTPAT) SU - to calculate a normalized antenna gain (antenna pattern factor) for a specified antenna elevation angle.

(2) Refractivity Initialization (REFINIT) SU - to check for valid environmental profile inputs and to initialize the refractivity arrays. This SU references the REMDUP SU of the PESTEP CSC.

(3) Trace for Minimum Angle (TRACEA) SU - to perform a ray trace to determine the minimum angle required (based on the reflected ray) in obtaining a PE solution for all heights up to the maximum output height (or the largest height allowed from the maximum transform size) and for all ranges beyond 90% of the maximum output range.

(4) Dielectric Initialization (DIEINIT) SU - to determine the conductivity and relative permittivity as a function of frequency in MHz based on general ground composition types.

(5) Get FFT Size (GETFFTSZ) SU - to determine the required transform size based on the maximum PE propagation angle and the specified maximum output height.

(6) Starter Field Initialization (XYINIT) SU - to calculate the complex PE solution at range zero. This SU references the GETALN SU of the PESTEP CSC.

(7) Fast-Fourier Transform (FFT) SU - to separate the real and imaginary components of the complex PE field into two real arrays and then to reference the SINFFT SU.

(8) Sine Fast-Fourier Transform (SINFFT) SU - to transform each portion of the PE solution.

(9) Trace Launch Angle (TRACEH) SU - to perform a ray trace for a single ray and store all heights traced to each output range step.

(10) Free-Space Propagator Phase Term (PHASE1) SU - to initialize the free-space propagator array for subsequent use in the PESTEP SU.

(11) Environmental Propagator Phase Term (PHASE2) SU - to calculate the environmental phase term for an interpolated environment profile.

(12) Profile Reference (PROFREF) SU - to adjust the current refractivity profile so that it is relative to a reference height.

(13) Interpolate Profile (INTPROF) SU - to perform a linear interpolation vertically with height on the refractivity profile.

Parabolic Equation Step (PESTEP) CSC - to advance the entire TPEM CSCI algorithm one output range step, referencing various SUs to calculate the propagation loss at the current output range. This CSC references the FFT SU and PHASE2 SU of the PEINIT CSC. The PESTEP CSC component SUs include:

(1) DOSHIFT SU - to shift the field by the number of bins, or PE mesh heights corresponding to local ground height.

(2) GETALN SU - to compute the impedance term in the Leontovich boundary condition, and the complex index of refraction for finite conductivity and vertical polarization calculations.

(3) Free Space Range Step (FRSTP) SU - to propagate the complex PE solution field in free space by one range step. This SU references the FFT SU of the PEINIT CSC.

(4) Refractivity Interpolation (REFINTER) SU - to interpolate both horizontally and vertically on the modified refractivity profiles. This SU references the PROFREF SU and INTPROF SU of the PEINIT CSC.

(5) Remove Duplicate Refractivity Levels (REMDUP) SU - to remove any duplicate refractivity levels in the currently interpolated profile.

(6) Calculate Propagation Loss (CALCLOS) SU - to determine the propagation loss at each output height point at the current output range.

(7) Get Propagation Factor (GETPFAC) SU - to determine the propagation factor at the specified height in dB.

4.2 Concept of Execution

The program flow of the TPEM CSCI is illustrated in Figure 4- 1. Note that the TPEM CSCI is shown within the context of a calling CSCI application such as one that generates a coverage or loss diagram. The efficient implementation of the TPEM CSCI will have far reaching consequences upon the design of an application CSCI beyond those mentioned in Section 7.3. For example, Figure 4- 1 shows checking for the existence of a previously created TPEM output file prior to the access of the TPEM CSCI. The application CSCI will have to consider if the atmospheric or terrain environment has changed since the TPEM output file was created or if any new height or range requirement is accommodated within the existing TPEM CSCI output file. Because these and many more considerations are beyond the scope of this document to describe, an application CSCI designer should work closely with the TPEM CSCI development agency in the implementation of the TPEM CSCI.

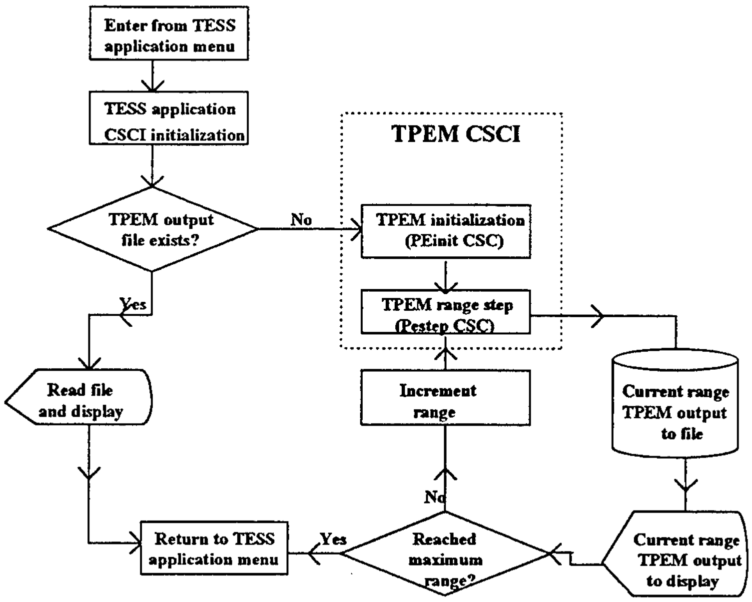

Figure 4- 1 Program flow of the TPEM CSCI

4.3 Interface Design

4.3.1 Interface Identification and Diagrams

The TPEM CSCI interface design consists of two FORTRAN INCLUDE files for the external and internal data interface, FORTRAN CALL statements for both output data and internal interfacing, and several FORTRAN COMMON blocks for the internal interface. The INCLUDE files are called TPEM.INC and FFTSIZ.INC. These INCLUDE statements provide several constants necessary for dimensioning of internal data arrays. The COMMON block names are: (1) ARRAYS, (2) HTVAR, (3) IMPEDANCE, (4) MISCVAR, (5) PARINIT, (6) PATTERN, (7) PEVAR, (8) PROFWREF, (9) RHSTPS, and (10) TRVAR.

4.3.2 External Interface

The TPEM CSCI is accessed, through the PEINIT CSC, by a subroutine call from the TESS CSCI which should provide, as global data elements, the values specified in Table 4-1 through Table 4-4.

The TPEM CSCI external data elements, i.e. those data which must be provided by the calling TESS CSCI in the INCLUDE file prior to the TPEM CSCI execution can be divided into four classifications. The first is external data related to the atmospheric environment, specified within Table 4-1; the second is data related to the EM system being assessed, specified within Table 4-2; the third is data related to the implementation of the TPEM CSCI by the TESS CSCI, specified within Table 4-3; and the fourth is data related to the terrain information, specified within Table 4-4. Each table lists the type, units, and bounds of each data element. Table 4-5 specifies the output data of the TPEM CSCI model passed back to the calling CSCI via the FORTRAN CALL statements.

Table 4-1 TPEM CSCI Environmental Data Element Requirements

Name	Description	Type	Units	Bounds
refmsl	Profile modified refractivity array referenced to mean sea level	real	M	≥ 0.0 [a]
hmsl	Profile height array	real	m	≥ 0.0 [a]
nprof	Number of profiles	integer	N/A	≥ 1
lvlep	Number of profile levels	integer	N/A	≥ 2
rngprof	Array of ranges to each profile	real	m	≥ 0.0

[a] Couplets of height and modified refractivity associated with that height are referred to within this document as an environmental profile.

Table 4-2 TPEM CSCI External EM System Data Element Requirements

Name	Description	Type	Units	Bounds
bwidth	Antenna vertical beam width	real	degree	0.5 to 45.0
elev	Antenna elevation angle	real	degree	-10.0 to 10.0
freq	EM system frequency	real	MHz	100.0 to 20,000.0
ipat	Antenna pattern 0 = Omni-directional 1 = Gaussian 2 = Sine (X)/X 3 = Cosecant-squared 4 = Generic height-finder	integer	N/A	0 to 4
polar	Antenna polarization H = Horizontal V = Vertical	character	N/A	'H' or 'V'
antht	Antenna height above local ground at range 0.0 m	real	m	≥ 1.0

Table 4-3 TPEM CSCI External Implementation Constants

Name	Description	Type	Units	Bounds
mxnfft	Maximum power of 2 for transform size	integer	N/A	≥ 9
maxpts	Maximum number points for FFT array	integer	N/A	$= 2^{mxnfft}$
mxlvls	Maximum number of profile levels for all possible applications of TPEM	integer	N/A	≥ 2 [b]
mxrout	Maximum number of range output points for all possible applications of TPEM	integer	N/A	≥ 1 [a]
nrout	Number of range output points for a particular application of TPEM	integer	N/A	= *mxrout* [a]
mxzout	Maximum number of height output points for all possible applications of TPEM	integer	N/A	≥ 1 [a]
nzout	Number of height output points for a particular application of TPEM	integer	N/A	= *mxzout* [a]
mxnprof	Maximum number of profiles for all possible applications of TPEM	integer	N/A	≥ 1 [b]
mxter	Maximum number of height/range points in terrain profile for all possible applications of TPEM	integer	N/A	≥ 2 [b]
lerr6	Logical flag to allow for error -6 to be bypassed	logical	N/A	'.true.' or '.false.' [a]
lerr12	Logical flag to allow for error -12 to be bypassed	logical	N/A	'.true.' or '.false.' [a]

Table 4-3 TPEM CSCI External Implementation Constants (con't)

Name	Description	Type	Units	Bounds
rmax	Maximum range output for a particular application of TPEM	real	m	$\geq$ 5000.0 [b]
hmin	Minimum height output for a particular application of TPEM	real	m	$\geq$ 0.0 [b]
hmax	Maximum height output for a particular application of TPEM	real	m	$\geq$ 100.0 [b]
propang	Maximum PE propagation angle	real	deg	$\geq$ 0.0 [a]

[a] refer to section 7.2 for a complete description.
[b] refer to section 7.3 for a complete description.

Table 4-4 TPEM CSCI External Terrain Data Element Requirements

Name	Description	Type	Units	Bounds
terx	Terrain profile range array	real	m	≥ 0.0 [a]
tery	Terrain profile height array	real	m	≥ 0.0 [a]
itp	Number of terrain profile points for a particular application of TPEM	integer	N/A	$\leq$ *mxter* [a]
igr	Number of ground types for a particular application of TPEM	integer	N/A	≤ 50
igrnd	Array of ground composition types for a particular application of TPEM 0 = Sea water 1 = Fresh water 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = User defined	integer	N/A	$0 \leq igrnd \leq 5$ [a]
rgrnd	Array of ranges for which ground types are applied for a particular application of TPEM	real	m	≥ 0.0 [a]
dielec	2-dimensional array of relative permittivity and conductivity for a particular application of TPEM	real	N/A	>0 [a]
	[a] refer to Section 7.3 for a complete description.			

Table 4-5 TPEM CSCI Output Data Element Requirements

Name	Description	Type	Units	Source
hminter	Minimum elevation height of terrain profile	real	m	PEINIT CSC
ierror	Integer value that is returned if an error exists in input data	integer	N/A	PEINIT CSC
mloss	Propagation loss	integer	cB	PESTEP CSC
jstart	Output height index at which valid propagation loss values begin	integer	N/A	PESTEP CSC
jend	Output height index at which valid propagation loss values end	integer	N/A	PESTEP CSC
rout	Current range	real	m	PESTEP CSC
	[a]Refer to Section 4.3.4 for a complete description.			

4.3.3 Internal Interface

Section 4.2 shows the relationship between the TPEM CSCI and its two main CSCs PEINIT and PESTEP. This relationship is illustrated in Figure 4- 1. The internal interface between these two CSCs and the TPEM CSCI is left to the design. However, the internal structure of the TPEM CSCI and its CSCs and SUs is shown in Table 4-6. The left two columns show the calling subroutines, and the right two columns the subroutines called. Columns 2 and 4 in Table 4-6 give the section number in Section 5 where more details about the various CSCs and SUs of the TPEM CSCI can be found.

Table 4-6 TPEM Internal Interface Design

Software Design Description		Software Design Description	
SDD Design Description Name	SDD Section Number	Software Design Description Name	SDD Section Number
Parabolic Equation Initialization (PEINIT) CSC	5.1	Refractivity Initialization (REFINIT) SU	5.1.2
Refractivity Initialization (REFINIT) SU	5.1.2	Remove Duplicate Refractivity Levels (REMDUP) SU	5.2.5
Parabolic Equation Initialization (PEINIT) CSC	5.1	Trace for Minimum Angle (TRACEA) SU	5.1.3
Parabolic Equation Initialization (PEINIT) CSC	5.1	Dielectric Initialization (DIEINIT) SU	5.1.4
Parabolic Equation Initialization (PEINIT) CSC	5.1	Get FTT Size (GETFFTSZ) SU	5.1.5
Parabolic Equation Initialization (PEINIT) CSC	5.1	Starter Field Initialization (XYINIT) SU	5.1.6
Starter Field Initialization (XYINIT) SU	5.1.6	Antenna Pattern (ANTPAT) SU	5.1.1
Starter Field Initialization (XYINIT) SU	5.1.6	GETALN SU	5.2.2
Parabolic Equation Initialization (PEINIT) CSC	5.1	Fast-Fourier Transform (FFT) SU	5.1.7
Fast-Fourier Transform (FFT) SU	5.1.7	Sine Fast-Fourier Transform (SINFFT) SU	5.1.8
Parabolic Equation Initialization (PEINIT) CSC	5.1	Trace Launch Angle (TRACEH) SU	5.1.9
Parabolic Equation Initialization (PEINIT) CSC	5.1	Free-Space Propagator Phase Term (PHASE1) SU	5.1.10
Parabolic Equation Initialization (PEINIT) CSC	5.1	Environmental Propagator Phase Term (PHASE2) SU	5.1.11

Table 4-6 TPEM Internal Interface Design (con't)

Software Design Description		Software Design Description	
SDD Design Description Name	SDD Section Number	Software Design Description Name	SDD Section Number
Parabolic Equation Initialization (PEINIT) CSC	5.1	Profile Reference (PROFREF) SU	5.1.12
Parabolic Equation Initialization (PEINIT) CSC	5.1	Interpolate Profile (INTPROF) SU	5.1.13
Parabolic Equation Step (PESTEP) CSC	5.2	DOSHIFT SU	5.2.1
Parabolic Equation Step (PESTEP) CSC	5.2	GETALN SU	5.2.2
Parabolic Equation Step (PESTEP) CSC	5.2	Free Space Range Step (FRSTP) SU	5.2.3
Free Space Range Step (FRSTP) CSC	5.2.3	Fast-Fourier Transform (FFT) SU	5.1.7
Parabolic Equation Step (PESTEP) CSC	5.2	Refractivity Interpolation (REFINTER) SU	5.2.4
Refractivity Interpolation (REFINTER) SU	5.2.4	Profile Reference (PROFREF) SU	5.1.12
Refractivity Interpolation (REFINTER) SU	5.2.4	Interpolate Profile (INTPROF) SU	5.1.13
Refractivity Interpolation (REFINTER) SU	5.2.4	Remove Duplicate Refractivity Levels (REMDUP) SU	5.2.5
Parabolic Equation Step (PESTEP) CSC	5.2	Calculate Propagation Loss (CALCLOS) SU	5.2.6
Calculate Propagation Loss (CALCLOS) SU	5.2.6	Get Propagation Factor (GETFAC) SU	5.2.7
Parabolic Equation Step (PESTEP) CSC	5.2	Fast-Fourier Transform (FFT) SU	5.1.7
Parabolic Equation Step (PESTEP) CSC	5.2	Environmental Propagator Phase Term (PHASE2) SU	5.1.11

4.3.4 Internal Data

The TPEM CSCI requires several internal arrays to be pre-dimensioned. Since the software documentation standards and coding requirements do not allow for dynamic dimensioning, these arrays must be dimensioned at program compile time. The implementation constants *mxrout* and *mxzout* refer to the extreme dimensions for various range and height related arrays for all possible applications of the TPEM CSCI and will be specified when the TPEM CSCI is compiled.

Due to the computational intensity of the TPEM CSCI, it may not be necessary or desirable to use the extreme capability of the TPEM CSCI for all applications. The variables *nrout* and *nzout* refer to the desired number of range and height output points for any one particular application, and will be specified when the PEINIT SU is called.

For the TPEM CSCI implementation within the TESS coverage and loss diagram applications, *mxrout* and *nrout* should be set equal, and *mxzout* and *nzout* should be set equal. The values chosen for *mxrout* and *mxzout* must be applicable to the coverage diagram.

One of the parameters returned to the TESS application from the PEINIT CSC is *ierror*. This is to allow for greater flexibility in how input data is handled within the TESS application. Table 4-7 lists all possible errors that can be returned.

Table 4-7 PEINIT SU Returned Error Definitions

error	**Definition**
-6	Last range in terrain profile is less than *rmax*. Will only return this error if *lerr6* set to '.true.'
-8	*hmax* is less than maximum height of terrain profile
-12	Range of last environment profile given (for range-dependent case) is less than *rmax*. Will only return this error if *lerr12* set to '.true.'
-14	Last gradient in any environment profile is negative
-17	Range points of terrain profile are not increasing
-18	First range value in terrain profile is not 0.

The logical variables *lerr6* and *lerr12*, when set to '.false.', allow the TESS application to bypass their associated errors as these are not critical to the operation of the TPEM CSCI.

The TPEM CSCI provides propagation loss for moderately low angles and heights. It does not provide propagation loss for *all* heights and ranges desired. Propagation loss values can be provided automatically at all heights from *at least* 90% of the desired maximum range, *rmax*, to *rmax*. At lesser ranges, propagation loss values will be reduced in height. If greater coverage is desired (i.e., propagation loss desired at closer ranges and larger heights), a non-zero positive value for *propang* must be specified. The TPEM CSCI can then provide propagation loss for all heights and ranges up to the maximum angle specified by *propang*. Set *propang* equal to zero to use the TPEM CSCI's default angle and coverage.

5. CSCI DETAILED DESIGN

A description of each component of the TPEM CSCI is provided in the following subsections together with the data flow and relationships between the main CSCs and sub level SUs.

5.1 Parabolic Equation Initialization (PEINIT) CSC

The purpose of the PEINIT SU is to interface with various SUs for the complete initialization of the TPEM CSCI. Figure 5- 1 illustrates the program flow for the PEINIT SU.

Upon entering the PEINIT SU, several variables are set. The error flag (*ierror*), the maximum PE propagation angle (Θ_{max}), the minimum reference height for internal calculations (y_{minter}), and the maximum tangent angle from the source to the terrain peaks (*angu*) are set to 0. The logical terrain flag (*fter*) is set to .FALSE., and the antenna height relative to the reference height (ant_{ref}) is set to $h_{transmitter}$.

Next, the maximum output range, x_{max}, the maximum output height, y_{max}, and the minimum output height, y_{min}, are checked for valid numerical values. x_{max}, is set to the value input from the calling CSCI or 5000 meters, whichever is greater; and y_{max}, is set to the value input from the calling CSCI or 100 meters, whichever is greater. If y_{min} is greater than y_{max}, then *ierror* is set to -42 and the PEINIT CSC is exited. If the maximum output range and minimum and maximum output height values are valid, then the PEINIT CSC proceeds to the next step.

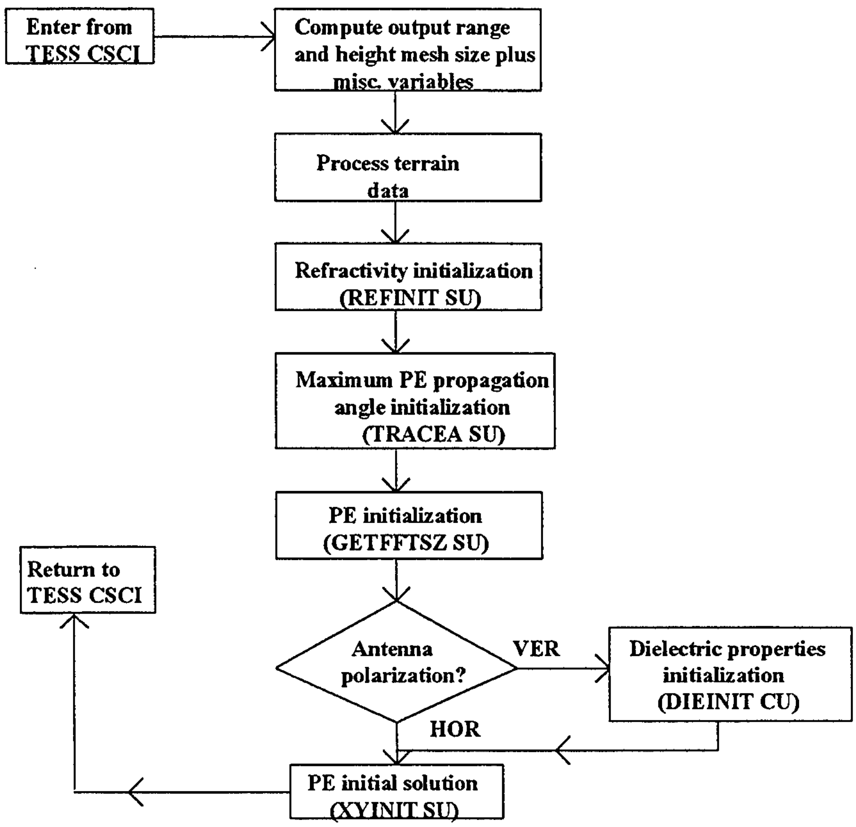

Figure 5- 1 Program flow of the PEINIT CSC

The atmospheric volume must be "covered" or resolved with a mesh of calculation points which will, as a matter of routine, exceed the height/range resolution requirements of the particular application of the TPEM CSCI. The height and range mesh size per TPEM CSCI output point, Δz_{out} and Δx_{out} respectively, are calculated from the number of TPEM outputs and the maximum range and height as follows:

$$\Delta x_{out} = \frac{x_{max}}{vnp.nrout} \tag{1}$$

$$\Delta z_{out} = \frac{\left(y_{max} - y_{min}\right)}{vnp.nzout} \,. \tag{2}$$

The array z_{out}, containing all output height points is determined by the formula

$$z_{out}(i) = y_{min} + i\,\Delta z_{out} \qquad \text{for } i = 1,2,3,\ldots vnp.nzout. \tag{3}$$

Next, several variables are determined for later calculations. These are the wavelength, λ, the free-space wavenumber, k_o, and the constant, con. They are determined as follows:

$$\lambda = \frac{c_o}{f_{MHz}} \tag{4}$$

where c_o is the speed of light (299.79×10^6 m/s);

$$k_o = \frac{2\pi}{\lambda}; \tag{5}$$

$$con = 10^6 \; k_o.$$

Next, the constants used to determine the antenna pattern factor are calculated as follows. First, the antenna vertical beamwidth is not allowed to go out of the range from 0.5 to 45.0 degrees and is determined from:

$$\mu_{bwr} = \text{AMIN1}\left(\text{AMAX1}\left\{sv.bwidth, .5\right\}, 45.0\right)\frac{\pi}{180.0} \,. \tag{6}$$

Then the antenna pattern elevation angle μ_{0r} is not allowed to go outside the range from -10 to 10 degrees and is determined from

$$\mu_{or} = \mathrm{AMIN1}\left(\mathrm{AMAX1}\left\{sv.elev, -10.0\right\}, 10.0\right)\frac{\pi}{180.0} . \qquad (7)$$

If the antenna pattern is Gaussian (i_{ptrn}=1), then ant_{fac} is given by

$$ant_{fac} = \frac{.34657359}{\mathrm{SIN}\left(\frac{\mu_{bwr}}{2}\right)}. \qquad (8)$$

If the antenna pattern is Sin(X)/X (i_{ptrn} = 2), or a generic height finder (i_{ptrn} = 4), then ant_{fac} is given by

$$ant_{fac} = \frac{1.39157}{\mathrm{SIN}\left(\frac{\mu_{bwr}}{2}\right)^2} . \qquad (9)$$

and μ_{max} is given by

$$\mu_{max} = \mathrm{TAN}^{-1}\left(\frac{\pi}{ant_{fac}\sqrt{1 - \left(\frac{\pi}{ant_{fac}}\right)^2}}\right). \qquad (10)$$

The terrain profile is initially examined and unnecessary range/height points are discarded if neighboring terrain slopes are redundant. The minimum terrain height is determined, and then the entire terrain profile is adjusted by this height so that all internal calculations are referenced to this height. This is done in order to maximize the PE transform calculation volume.

First, *fter* is set to .TRUE. if *tr.itp* is greater than 0. Next, the terrain profile is checked for increasing range points. If any range point is not increasing, then *ierror* is set to -17 and the PEINIT CSC is exited. The first range value is also checked; if it is not 0 then, *ierror* is set to -18 and the PEINIT CSC is exited. Next, the value of the last range

point is checked. If it is less than x_{max} and *ef.lerr6* is set to .TRUE., then *ierror* is set to -6 and the PEINIT CSC is exited. If the last range value is less than x_{max} and *ef.lerr6* is set to .FALSE., then an extra terrain point is added with range equal to x_{max}, and height equal to the last terrain height point.

A scratch file is then opened for temporary storage of terrain points. Only those terrain points/segments whose surrounding slope difference is greater than 10^{-3} are stored. The slope difference *scd* is computed by

$$scd = \left[\frac{tr.tery(i+1) - tr.tery(i)}{\text{AMAX1}\left(10^{-3}, tr.terx(i+1) - tr.terx(i)\right)}\right] - \left[\frac{tr.tery(i) - tr.tery(i-1)}{\text{AMAX1}\left(10^{-3}, tr.terx(i) - tr.terx(i-1)\right)}\right] . \quad (11)$$

Once all desired terrain points are stored, they are read back into arrays *tr.terx* and *tr.tery*. Next, the minimum (y_{minter}) and maximum (y_{termax}) terrain heights of the profile are determined. The antenna height relative to the local ground height at range 0 (ant_{ref}) is given by

$$ant_{ref} = h_{transmitter} + tr.tery(1) . \quad (12)$$

Next, the slopes along the entire terrain profile, *slp*, is computed along with the maximum tangent angle from the source to the terrain peaks, *angu*. These quantities are given by

$$slp(i) = \frac{tr.tery(i+1) - tr.tery(i)}{\text{AMAX1}\left\{10^{-3}, tr.terx(i+1) - tr.terx(i)\right\}} \qquad \text{for } i = 1,2,3\ldots\{tr.itp - 1\} \quad (13)$$

$$angu = \text{AMAX1}\left[angu, \text{TAN}^{-1}\left(\frac{tr.tery(i) - antref}{tr.terx(i)}\right)\right] + .5 \text{ deg} \qquad \text{for } i = 1,2,3\ldots\{tr.itp - 1\} . \quad (14)$$

The following variables used in later computations are initialized. The minimum height relative to y_{minter} (y_{mref}), the maximum calculation height relative to y_{minter} (ht_{lim}), and z_{lim} are determined by

$$y_{mref} = y_{min} - y_{minter} \quad (15)$$

$$ht_{lim} = y_{max} - y_{minter} \quad (16)$$

$$z_{\lim} = \text{AMAX1}\left(ht_{\lim}, ant_{ref}\right). \tag{17}$$

Next, a REFINIT SU is referenced to initialize the TESS CSCI specified modified refractivity and also to test for valid environment profiles. Once the refractivity profiles are initialized, the gradients $(\partial M(j)/\partial h)$ of the refractivity profile at range 0 are computed as follows:

$$grad_j = \frac{refdum(j+1) - refdum(j)}{htdum(j+1) - htdum(j)} \qquad \text{for } j = 1,2,3,\ldots\{lvlep - 1\} \tag{18}$$

$$\begin{aligned} \frac{\partial M(j)}{\partial h} &= 10^6 \text{ AMAX1}\left[grad_j, 10^{-3}\right] \qquad \text{if } grad_j > 0 \\ \frac{\partial M(j)}{\partial h} &= 10^6 \text{ AMIN1}\left[grad_j, -10^{-3}\right] \qquad \text{if } grad_j < 0 \end{aligned} \quad . \tag{19}$$

In order to determine the critical angle, a_{crit}, the modified refractivity at the source height, M_o, and the minimum refractivity above (M_a) and below (M_b) the source height are computed. The variable a_{crit} is then given by

$$a_{crit} = \text{AMAX1}\left(\alpha_1, \alpha_2\right), \tag{20}$$

where

$$\alpha_1 = \sqrt{2 \times 10^{-6} \text{AMAX1}\left(0, M_o - M_a\right)} \tag{21}$$

$$\alpha_2 = \sqrt{2 \times 10^{-6} \text{ AMAX1}\left(0, M_o - M_b\right)} \ . \tag{22}$$

a_{crit} is now used to initialize Θ_{max}, with x_{lim} set equal to 90% of x_{max}, and p_{angle} set equal to *vnp.propang* in radians. If p_{angle} is non-zero, then Θ_{max} is set equal to p_{angle}.

In order to automatically determine the maximum PE calculation angle (Θ_{max}), the TRACEA SU is referenced. This will determine, via ray tracing, the minimum angle for which adequate coverage can be given with the specified terrain and environment profile. Once a value for Θ_{max} is obtained from the TRACEA SU reference call, Θ_{max} is then divided by .75 since the upper ¼ of the calculation domain is filtered.

Due to numerical constraints, Θ_{max} is not allowed to go beyond certain values, depending on frequency. Θ_{max} is then checked to make sure it falls within the specified angular limits, based on frequency as follows:

$$\Theta_{max} = \text{AMAX1}\left(4^\circ, \Theta_{max}\right) \quad \text{for } f_{MHz} \leq 200 \tag{23}$$

$$\Theta_{max} = \text{AMAX1}\left(3^\circ, \Theta_{max}\right) \quad \text{for } 200 < f_{MHz} \leq 400 \tag{24}$$

$$\Theta_{max} = \text{AMAX1}\left(2^\circ, \Theta_{max}\right) \quad \text{for } 400 < f_{MHz} \leq 600 \tag{25}$$

$$\Theta_{max} = \text{AMAX1}\left(1^\circ, \Theta_{max}\right) \quad \text{for } 600 < f_{MHz} < 1500 \qquad 26)$$

$$\Theta_{max} = \text{AMAX1}\left(.9^\circ, \Theta_{max}\right) \quad \text{for } 1500 \leq f_{MHz} < 2500 \tag{27}$$

$$\Theta_{max} = \text{AMAX1}\left(.8^\circ, \Theta_{max}\right) \quad \text{for } 2500 \leq f_{MHz} < 2900 \tag{28}$$

$$\Theta_{max} = \text{AMAX1}\left(.7^\circ, \Theta_{max}\right) \quad \text{for } 2900 \leq f_{MHz} < 4100 \tag{29}$$

$$\Theta_{max} = \text{AMAX1}\left(.6^\circ, \Theta_{max}\right) \quad \text{for } 4100 \leq f_{MHz} < 5000 \tag{30}$$

$$\Theta_{max} = \text{AMAX1}\left(.5^\circ, \Theta_{max}\right) \quad \text{for } 5000 \leq f_{MHz} \leq 9000 \tag{31}$$

and if f_{MHz} is greater than 9000 MHz, then the value for Θ_{max} is left untouched and no lower limit is applied. If performing a vertical polarization case (i.e., *sv.polar* = 'V'), then Θ_{max} is doubled.

A GETFFTSZ SU is referenced to determine the fast Fourier transform (FFT) size for the calculated angle and to initialize data elements within the PE region which are dependent on the size of the FFT. The minimum size for the FFT is determined from the Nyquist criterion. Once the necessary PE parameters have been determined based on Θ_{max} and the maximum output height y_{max}, Θ_{max} is then "maximized" within the calculated z_{max} and FFT size. This is done only if *fter* is .TRUE. and if *sv.propang* is equal to 0.

Next, the radar horizon range in meters, r_{hor}, for the source height ($h_{transmitter}$) and 0 receiver height, is computed by

$$r_{hor} = 4124.5387 \sqrt{h_{transmitter}} \quad ; \tag{32}$$

and the initial PE range step is given by

$$\Delta x_{PE} = 2\, k_o\, \Delta z_{PE}^2 \quad . \tag{33}$$

Due to numerical constraints, numerical limits will be imposed on the PE range step, depending on y_{max} as follows. If performing a terrain case, then

$$\Delta x_{PE} = \text{AMAX1}\left(75,\, \Delta x_{PE}\right) \qquad \text{for } 5 \text{ km} \le y_{max} < 10 \text{ km} \tag{34}$$

$$\Delta x_{PE} = \text{AMAX1}\left(90,\, \Delta x_{PE}\right) \qquad \text{for } 10 \text{ km} \le y_{max} < 15 \text{ km} \tag{35}$$

$$\Delta x_{PE} = \text{AMAX1}\left(100,\, \Delta x_{PE}\right) \qquad \text{for } 15 \text{ km} \le y_{max} < 20 \text{ km} \tag{36}$$

$$\Delta x_{PE} = \text{AMAX1}\left(110,\, \Delta x_{PE}\right) \qquad \text{for } 20 \text{ km} \le y_{max} < 30 \text{ km} \tag{37}$$

$$\Delta x_{PE} = \text{AMAX1}\left(175,\, \Delta x_{PE}\right) \qquad \text{for } 30 \text{ km} \le y_{max} < 50 \text{ km} \tag{38}$$

$$\Delta x_{PE} = \text{AMAX1}\left(200,\, \Delta x_{PE}\right) \qquad \text{for } 50 \text{ km} \le y_{max} < 75 \text{ km} \tag{39}$$

$$\Delta x_{PE} = \text{AMAX1}\left(250,\, \Delta x_{PE}\right) \qquad \text{for } 75 \text{ km} \le y_{max} < 100 \text{ km} \tag{40}$$

$$\Delta x_{PE} = \text{AMAX1}\left(300,\, \Delta x_{PE}\right) \qquad \text{for } 100 \text{ km} \le y_{max} \quad . \tag{41}$$

An absolute upper limit of 700 m is imposed for Δx_{PE}. If not performing a terrain case, then Δx_{PE} is given as follows:

$$\Delta x_{PE} = \text{AMAX1}\left(30,\, \text{AMIN1}\left\{1000.,\, \Delta x_{PE}\right\}\right) \qquad \text{for } y_{max} < r_{hor} \quad ; \tag{42}$$

$$\Delta x_{PE} = \text{AMAX1}\left(300,\, \text{AMIN1}\left\{1000.,\, \Delta x_{PE}\right\}\right) \qquad \text{for } y_{max} \ge r_{hor} \quad . \tag{43}$$

Several variables are then initialized for later use in PE calculations. One-half of the PE range step is given by $\Delta x_{PE2} = \frac{1}{2}\ \Delta x_{PE}$; *plcnst*, used in calculating propagation loss, is determined as

$$plcnst = 20\ \text{ALOG10}\left(2\, k_o\right) \quad ; \tag{44}$$

the angle (or p-space) mesh size, Δp, the Fourier transform normalization constant, f_{norm}, the constant used in determining the free-space phase factors, *cnst*, the transform size minus 1, *nm1*, ¾ of the transform size, $n_{3/4}$, and twice the height (or z-space) mesh size, are determined as follows:

$$\Delta p = \frac{\pi}{z_{max}}; \qquad f_{norm} = \frac{2}{n_{fft}}; \tag{45}$$

$$cnst = \frac{\Delta p}{k_o}; \qquad nm1 = n_{fft} - 1; \tag{46}$$

$$n_{3/4} = \frac{3}{4} n_{fft}; \qquad \Delta z_{PE2} = 2\ \Delta z_{PE} \tag{47}$$

Next, the filter array, *FILT*, for subsequent filtering of the PE field, is given by

$$FILT(i) = \frac{1}{2} + \frac{1}{2} \cos\left[\frac{4\pi i}{n_{fft}}\right] \qquad \text{for } i = 0,1,2,\ldots \frac{n_{fft}}{4}. \tag{48}$$

If no ground constants were specified (i.e., $tr.i_{gr}$ is equal to 0), then the initial value for the ground constant parameters, *tr.igrnd* and *tr.rgrnd* are set to 0 and *tr.igr* is set to 1. If vertical polarization is specified, the DIEINIT SU is used to initialize the dielectric ground constants. For general ground types, the permittivity and conductivity are calculated as a function of frequency from curve fits to the permittivity and conductivity graphs shown in recommendations and reports of the International Radio Consulting Committee (Ref. d).

A PE starting SU (XYINIT) and an antenna pattern factor SU (ANTPAT) are referenced by the XYINIT SU to generate a first solution to the PE, along with a GETALN SU to compute the complex index of refraction and other variables used in the mixed transform method for vertical polarization. A FFT SU is referenced to transform the PE's starting solution from p- (or angle-) space, to z- (or height-) space.

Then, if vertical polarization is specified, variables used in Kuttler and Dockery's mixed transform method (Ref. h) are initialized. These are given by

$$C_1 = R_K \sum_{j=0}^{n_{fft}}{}' U_j \; R_T^j \tag{49}$$

$$C_2 = R_K \sum_{j=0}^{n_{fft}}{}' U_j \; \left(-R_T\right)^j \tag{50}$$

where the prime on the summation indicates that the values at the endpoints, 0 and n_{fft}, are to be weighted by a factor of ½.

Next, the variables x_{out}, y_{cur}, y_{curm}, and y_{last} are set equal to 0. If performing a terrain case, then the variable y_{last} is set equal to the ground height at the source. The height mesh array, is next defined by

$$ht = \text{FLOAT}\left(i\right) \Delta z_{PE} \qquad \text{for } i = 0,1,2,\ldots,n_{fft} \; . \tag{51}$$

The TRACEH SU is referenced to determine the height at each output range step below which valid propagation loss solutions will be valid. No propagation loss solutions are provided above these heights.

Finally, the PHASE1 SU should be referenced to initialize the free space propagator arrays. For a range-independent environment profile, the PROFREF SU adjusts the environment profiles by the internal reference height. If a smooth surface case is specified, and for a range-independent environment profile, the INTPROF SU is referenced to define the modified refractivity at all PE vertical mesh points, and the PHASE2 SU is referenced to initialize the environment propagator arrays.

Table 5-1 and Table 5-2 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the PEINIT CSC.

Table 5-1 PEINIT CSC Input Data Element Requirements

Name	Description	Units	Source
ef	Error flag structure for external implementation constants	N/A	TPEM.INC
ef.lerr6.	Element of user-provided error flag structure *ef* that will trap on certain errors if set to .TRUE	N/A	TPEM.INC
ef.lerr12	Element of user-provided error flag structure *ef* that will trap on certain errors if set to .TRUE	N/A	TPEM.INC
f_{MHz}	Frequency	MHz	TPEM.INC
$h_{transmitter}$	Transmitting antenna height above the local ground	meters	TPEM.INC
i_{ptrn}	Type of antenna pattern desired	N/A	TPEM.INC
maxn4	*maxpts* divided by 4; specifies the length of the filter array	N/A	TPEM.INC
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	TPEM.INC
mxlvls	Maximum number of height/M-unit levels	N/A	TPEM.INC
mxnfft	Maximum power of 2 for transform size	N/A	TPEM.INC
mxrout	Maximum number of output range points	N/A	TPEM.INC
mxter	Maximum number of height/range points allowed for terrain profile	N/A	TPEM.INC
mxzout	Maximum number of output height points	N/A	TPEM.INC
rf	Refractivity structure for external environmental data elements	N/A	TPEM.INC
rf.hmsl	2-dimensional array containing heights with respect to mean sea level of each profile. Array format must be *hmsl(i,j)* = height of *i*th level of jth profile. *j*=1 for range-independent cases	meters	TPEM.IN
rf.lvlep	Number of levels in refractivity profile	N/A	TPEM.IN
rf.nprof	Number of profiles	N/A	TPEM.IN
rf.refmsl	2-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be *refmsl(i,j)* = M-unit at *i*th level of *j*th profile. j=1 for range-independent cases	M-units	TPEM.IN
rf.rngprof	Ranges of each profile. *rngprof(i)* = range of *i*th profile	meters	TPEM.IN
sv	System structure for external system data elements	N/A	TPEM.INC

Table 5-1 PEINIT CSC Input Data Element Requirements (con't)

Name	Description	Units	Source
sv.bwidth	Half power (3 dB) antenna pattern beamwidth	degrees	TPEM.INC
sv.elev	Antenna pattern elevation angle	degrees	TPEM.INC
sv.polar	Character string indicating polarization	N/A	TPEM.INC
tr	Terrain structure for external terrain data elements	N/A	TPEM.INC
tr.dielec	2-dimensional array containing the relative permittivity and conductivity for user defined terrain	$^{\alpha}$	TPEM.INC
$tr.i_{gr}$	Number of different ground types specified	N/A	TPEM.INC
tr.igrnd	Type of ground composition for given terrain profile	N/A	TPEM.INC
tr.itp	Number of points in terrain profile	N/A	TPEM.INC
tr.rgrnd	Ranges at which the ground types apply	meters	TPEM.INC
tr.terx	Range points of terrain profile	meters	TPEM.INC
tr.tery	Height points of the terrain profile	meters	TPEM.INC
vnp	*INPUTVAR* structure for external implementation constants	N/A	TPEM.INC
vnp.nrout	Integer number of output range points desired	N/A	TPEM.INC
vnp.nzout	Integer number of output height points desired	N/A	TPEM.INC
x_{max}	Maximum output range	meters	TPEM.INC
y_{max}	Maximum output height	meters	TPEM.INC
y_{min}	Minimum output height	meters	TPEM.INC

$^{\alpha}$ conductivity has units of S/m.

Table 5-2 PEINIT CSC Output Data Elements

Name	Description	Units	Source
ierror	Integer value that is returned if any errors exist in input data	N/A	PEINIT CSC
x_{out}	Output range	meters	PEINIT CSC
y_{minter}	Reference height for internal calculations of the field U (minimum height of terrain profile)	meters	PEINIT CSC

5.1.1 Antenna Pattern (ANTPAT) SU

The purpose of the ANTPAT SU is to calculate an antenna pattern factor (normalized antenna gain), $f(\alpha)$, for a specified antenna elevation angle, α. Currently, antenna pattern factors are included for five types of antennas. These patterns include an omni-directional (i_{patrn}=0) type, a Gaussian (i_{patrn}=1) type, a Sin(X)/X (i_{ptrn} =2) type, a cosecant-squared (i_{ptrn} =3) type, and a generic height-finder (i_{ptrn} =4) type.

From the antenna factor (calculated in the PEINIT CSC) ant_{fac}, the antenna beam width μ_{bwr}, the antenna pattern elevation angle μ_{0r}; a specified elevation angle α for which the antenna pattern factor is desired; and the antenna radiation pattern type i_{ptrn}; the antenna factor is calculated as follows. If the antenna pattern is omni-directional (i_{ptrn}=0), then, the antenna pattern factor $f(\alpha)$, is given by

$$f(\alpha) = 1 \ . \tag{52}$$

If the antenna pattern is Gaussian (i_{ptrn}=1); then $f(\alpha)$, the antenna pattern factor, is given by

$$f(\alpha) = e^{\left(-ant_{fac}\left\{SIN[\alpha] - SIN[\mu_{0r}]\right\}^2\right)} \ . \tag{53}$$

If ($i_{ptrn} > 1$),the elevation angle relative to the antenna elevation angle, α_{pat}, is computed by

$$\alpha_{pat} = \alpha - \mu_{0r} \ . \tag{54}$$

If the antenna pattern is a Sin(X)/X) (i_{ptrn} =2) or a generic height-finder (i_{ptrn} =4), the following calculations are made:

(1) The radiation pattern is simulated as a Sin(X)/X type with the elevation angle relative to the antenna elevation angle, α_{pat}, adjusted to account for the current pointing angle of the main beam. Initially, the value for α_{pat} computed in the above equation is used.

(2) If the antenna radiation pattern type is a generic height-finder (i_{ptrn} =4), then for $|\mathrm{SIN}(\alpha)| > \mu_{or}$, α_{pat} is replaced by

$$\alpha_{pat} = \alpha - \mathrm{ABS}(\mathrm{SIN}\{\alpha\}) \ . \tag{55}$$

(3) The antenna pattern for both (i_{ptrn} =2 or i_{ptrn} =4) is now given as

$$f(\alpha) = 1. \qquad \text{for } |\alpha_{pat}| < 10^{-6} \ , \tag{56}$$

$$f(\alpha) = 0.03 \qquad \text{for } |\alpha_{pat}| > \mu_{max} \ , \tag{57}$$

otherwise the antenna factor is given as

$$f(\alpha) = \mathrm{AMIN1}\left(1., \mathrm{AMAX1}\left\{0.03, \frac{\mathrm{SIN}\left[ant_{fac}\,\mathrm{SIN}(\alpha_{pat})\right]}{ant_{fac}\,\mathrm{SIN}\left[\alpha_{pat}\right]}\right\}\right) \ . \tag{58}$$

If the antenna pattern is cosecant-squared (i_{ptrn} =3), the antenna factor is given as

$$f(\alpha) = \frac{\mathrm{SIN}(\mu_{bwr})}{\mathrm{SIN}(\alpha_{pat})} \qquad \text{for } \alpha_{pat} > \mu_{bwr} \ ; \tag{59}$$

$$f(\alpha) = \mathrm{AMIN1}\left(1., \mathrm{AMAX1}\left\{0.03, \left[1. + \frac{\alpha_{pat}}{\mu_{bwr}}\right]\right\}\right) \qquad \text{for } \alpha_{pat} < 0.0 \tag{60}$$

$$f(\alpha) = 1. \qquad \text{otherwise for } \alpha_{pat} > 0.0 \tag{61}$$

Table 5-3 and Table 5-4 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the ANTPAT SU.

Table 5-3 ANTPAT SU Input Data Element Requirements

Name	Description	Units	Source
ant_{fac}	Antenna pattern parameter (depends on i_{patrn} and μ_{bwr})	radian	PEINIT CSC
i_{ptrn}	Antenna pattern type	N/A	XYINIT SU
μ_{0r}	Antenna pattern elevation angle	radians	PEINIT CSC
μ_{bwr}	Antenna vertical beam width	radians	PEINIT CSC
μ_{max}	Limiting angle for SIN(X)/X and generic height finder antenna pattern factors	radians	PEINIT CSC
$\mathrm{SIN}(\mu_{0r})$	Sine of antenna elevation angle	N/A	XYINIT SU
$\mathrm{SIN}(\mu_{bwr})$	Sine of antenna vertical beam width	N/A	PEINIT CSC
$\mathrm{SIN}(\alpha)$	Sine of specified elevation angle	N/A	PEINIT CSC

Table 5-4 ANTPAT SU Output Data Element Requirements

Name	Description	Units	Source
$f(\alpha)$	Antenna pattern factor for specified elevation angle α		ANTPAT SU

5.1.2 Refractivity Initialization (REFINIT) SU

The purpose of the REFINIT SU is to check for valid environmental profile inputs and to initialize the refractivity arrays.

Upon entering, the maximum height h_{large} at which the refractivity profile is extrapolated is set to 10^6 meters and the maximum range r_{large} at which the refractivity profile is extrapolated is set to 10^{10} meters, respectively, in a DATA statement. In addition, *ierror* is initialized to zero.

Then the environmental data is checked for a range-dependent profile (i.e., the number of profiles, *rf.nprof*, is tested to determine whether there are more than one profile), and tested whether the range of the last profile entered, *rf.rngprof(rf.nprof)*, is less than the maximum output range specified vr_{max}. If so, an error message is returned (i.e., *ierror* is set equal to minus twelve), depending on the value of error flag, *elerr12*, set in the TESS CSCI itself.

The REFINIT SU tests for valid refractivity level entries for each profile. First, an extra level is added to the tabulated profiles with an extrapolated gradient (i.e., *rf.lvlep* is set equal to *rf.lvlep* plus one). A DO loop and a DO WHILE loop nested within it are used to test for profiles that contain multiple height/M-unit values that are equal. The DO loop index *i* runs from 1 through *rf.nprof*. The height difference h_{dif} between the last two differing height levels in each refractivity profile is initialized to zero in the beginning of the DO loop. Before the DO WHILE loop is entered, both the last user-specified level in the refractivity profile, *lvlm1*, and the second-to-last user-specified level in the refractivity profile, *lvlm2*, are both set equal to *rf.lvlep*, the number of levels in the refractivity profile. Then just within the DO WHILE loop, *lvlm1* is set to *lvlm1* minus one (i.e., *rf.lvlep*-1) and *lvlm2* is set to *lvlm1* minus one, (i.e., *rf.lvlep*-2). Using these indices h_{dif} is found from

$$h_{dif} = rf.hmsl(lvlm1,i) - rf.hmsl(lvlm2,i) \tag{62}$$

where the variable *rf.hmsl* is a 2-dimensional array containing heights with respect to mean sea level of each profile. The DO WHILE loop is executed until h_{dif} is greater than 10^{-6} (i.e., basically non-zero). After the DO WHILE loop is exited, the gradient in the refractivity profile, *grad*, is determined from

$$grad = \frac{ref.refmsl(lvlm1,i) - rf.refmsl(lvlm2,i)}{h_{dif}} \tag{63}$$

where the variable *rf.refmsl* is a 2-dimensional array containing refractivity with respect to mean sea level of each profile. If the last gradient, *grad*, in any profile is negative, the REFINIT SU returns a value of *ierror* equal to -14 and REFINIT SU is exited. If no errors are detected, the REFINIT SU then extrapolates the environmental profiles vertically to 1000 km in height as follows

$$\begin{aligned} &rf.hmsl(rf.lvlep,i) = h_{large} \\ &rf.refmsl(rf.lvlep,i) = rf.refmsl(lvlm1,i) + grad\left[h_{large} - rf.hmsl(lvlm1,i)\right] \end{aligned} . \tag{64}$$

The results of the extrapolation of the first environmental profile (i.e., the profile at range 0) vertically to 1000 km in height, *rf.hmsl* and *rf.refmsl*, are transferred to dummy arrays, *htdum* and *refdum*, respectively. First, the index i_s, the counter for the current profile, is initialized to one. Then the range of the next refractivity profile, *rv2*, is set equal to *rf.rngprof*(i_s). In a DO loop with index *i* running from 1 through *rf.lvlep*, this transfer is made as follows:

$$\begin{aligned} &refdum(i) = rf.refmsl(i,i_s) \\ &htdum(i) = rf.hmsl(i,i_s) \end{aligned} . \tag{65}$$

The last provided environment profile is duplicated at 10^7 km in range in the following manner. First, the index representing the final number of refractivity profiles, n_p, is set equal to *rf.nprof* plus one. .The variable *rf.rngprof*(n_p) is set equal to the value of r_{large} which has a value of 10^7 km. Then the profile is duplicated at r_{large} in a DO loop with index *i* running from 1 to *rf.lvlep* as follows:

$$\begin{aligned} &rf.hmsl(i,n_p) = rf.hmsl(i,n_p - 1) \\ &rf.refmsl(i,n_p) = rf.refmsl(i,n_p - 1) \end{aligned} . \tag{66}$$

Finally, the number of height/refractivity levels in the profile, *lvlep*, is set equal to *rf.lvlep*, and a reference is made to the REMDUP SU to remove any duplicate refractivity levels in the currently interpolated profile.

Table 5-5 and Table 5-6 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the REFINIT SU.

Table 5-5 REFINIT SU Input Data Element Requirements

Name	Description	Units	Source
elerr12	Element of user-provided error flag structure *ef* that will trap on certain errors if set to .TRUE	N/A	PEINIT CSC
mxlvls	Maximum number of height/M-unit levels	N/A	TPEM.INC
rf	Refractivity structure for external environmental data elements	N/A	TPEM.INC
rf.hmsl	2-dimensional array containing heights with respect to mean sea level of each profile. Array format must be *hmsl(i,j)* = height of *i*th level of jth profile. *j*=1 for range-independent cases	meters	TPEM.INC
rf.lvlep	Number of levels in refractivity profile	N/A	TPEM.INC
rf.nprof	Number of profiles	N/A	TPEM.INC
rf.refmsl	2-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be *refmsl(i,j)* = M-unit at *i*th level of *j*th profile. j=1 for range-independent cases	M-units	TPEM.INC
rf.rngprof	Ranges of each profile. *rngprof(i)* = range of *i*th profile	meters	TPEM.INC
vr_{max}	Maximum range	meters	PEINIT CSC

Table 5-6 REFINIT SU Output Data Element Requirements

Name	Description	Units	Source
htdum	Dummy array containing height values for current (horizontally interpolated) profile	meters	REFINIT SU
ierror	Integer value that is returned if any errors exist in input data	N/A	REFINIT SU
i_s	Counter for current profile	N/A	REFINIT SU
lvlep	Number of height/refractivity levels in profile	N/A	REFINIT SU
refdum	Dummy array containing refractivity values for current (horizontally interpolated) profile	M-units	REFINIT SU
rv2	Range of the next refractivity profile	meters	REFINIT SU

5.1.3 Trace for Minimum Angle (TRACEA) SU

The purpose of the TRACEA SU is to perform a ray trace to determine the minimum angle required (based on the reflected ray) in obtaining a PE solution for all heights up to the maximum output height (or the largest height allowed from the maximum transform size) and for all ranges up to 90% of the maximum output range. The maximum PE propagation angle, Θ_{max}, is then determined from this angle for smooth surface and automatic angle calculation.

For terrain cases, the maximum PE propagation angle, Θ_{max}, will have already been set to the larger of the critical angle (if a duct exists), the angle that clears the highest terrain peak, or the tangent angle determined from the maximum output height and the maximum output range.

If a maximum propagation angle is specified by the TESS CSCI or another calling CSCI, then the maximum PE propagation angle is determined based on this given angle. However, a ray trace is still performed in order to determine the initial launch angle such that the local angle of the ray remains less than the specified maximum propagation angle. The initial launch angle is to be used in the TRACEH SU described later.

First, a constant Θ_{15} is defined in a DATA statement as 15 degrees in radians. Then the following internal functions are defined.

$$\mathrm{RADA1}(a,b) = a^2 + 2\ grad\ b \tag{67}$$

$$\mathrm{RP}(a,b) = a + \frac{a}{grad} \tag{68}$$

$$\mathrm{AP}(a,b) = a + b\ grad \tag{69}$$

$$\mathrm{HP}(a,b,c) = a + \frac{b^2 - c^2}{2\ grad} \tag{70}$$

Initially, the internal variable for the starting launch angle, a_s, is set to the negative of the input maximum propagation angle, Θ_{max}, used in the PE calculation. The index, I_{dn}, used to increment or to decrement the initial launch angle is set to -1.

For the terrain case, a_s is set to the value of Θ_{max}. In addition, if the user specified propagation angle, p_{angle}, in radians is less than or equal to 10^{-6} (i.e., for all intents and purposes, a zero angle and therefore, automatic calculation of Θ_{max} will be performed); then I_{dn} is set to 1. Further, if p_{angle}, is greater than or equal to 10^{-6} and the slope of the first segment of terrain, *slp*(1), is less than or equal to 10^{-6} (i.e., terrain is initially flat and non-zero user defined maximum propagation angle); then the range at which the ray is reflected, r_{ref}, is given by

$$r_{ref} = \frac{ant_{ref} - tr.tery(1)}{\mathrm{TAN}(\Theta_{max})} \tag{71}$$

where ant_{ref} is the transmitting antenna height relative to the reference height y_{minter}, and *tr.tery*(1) is the height point of the terrain profile in meters. If r_{ref} so determined is less than the range of the second terrain point *tr.terx*(2), then this case is treated as if this were a smooth surface problem. This results in a_s being set to the negative of Θ_{max} and in I_{dn} being set to -1. Finally, the variables a_{sl} (last starting launch angle value), a_{mxcurl} (last maximum of local angle along ray), and i_{set} (flag to test whether or not to stop loop used to determine the launch angle) are all set to zero.

A FORTRAN DO WHILE statement loop is used to determine the launch angle Θ_{luanch}. The loop is executed until the variable i_{set} has a non-zero value. A second, nested, DO WHILE loop is used perform a ray trace of the ray until the ray has reached the height limit for the ray trace, z_{lim}, and/or x_{lim} (90% of the maximum output range x_{max}). The second loop is executed until the logical variable *loop* is false. Prior to the first execution of the second DO WHILE loop, several variables are initialized or incremented. First, a_s is increased or decreased by 1 milliradian, depending on the value of I_{dn}. Then, the angle of the ray before the trace step, a_0, is set to the value of a_s. The variables; a_{mxcur} (maximum of local angle along ray), x_0 (range of ray before trace step), x_{PE} (range at which valid loss values will begin to be calculated); are all set to zero. The height of the ray before the trace step, h_0, is set equal to the transmitting antenna height relative to the reference height y_{minter}. The index, j_l, of the current refractivity layer the ray is tracing through is set equal to the index, j_{ls}, of the refractivity array at which the antenna height is located. Finally, the logical variable *loop* is set to true.

The processing of the inner DO WHILE loop is as follows. First, *grad* is set equal to the gradient of the first profile at j_l, $\partial M(j_l)/\partial h$. Depending on the value of a_0, the value of the height of the ray after the trace step, h_1, is determined as follows

$$\begin{aligned}
h_1 &= htdum(j_l) && for \;\; a_0 < 0.0 \\
h_1 &= htdum(j_l+1) && for \;\; a_0 > 0.0 \\
h_1 &= htdum(j_l) && for \;\; a_0 = 0.0 \;\; and \;\; grad < 0.0 \\
h_1 &= htdum(j_l+1) && for \;\; a_0 = 0.0 \;\; and \;\; grad > 0.0
\end{aligned} \quad . \tag{72}$$

If the value of h_1 so determined is greater than z_{lim}, then h_1 is set equal to z_{lim}. Next, the radical, *rad* is found from

$$rad = \mathrm{RADA1}(a_0, h_1 - h_0) \; . \tag{73}$$

If *rad* is greater than 0.0, then the angle of the ray after the trace step, a_1, is given by

$$a_1 = \mathrm{SIGN}(1.0, a_0) \; \mathrm{SQRT}(rad) \; . \tag{74}$$

If *rad* is less than or 0.0, then a_1 and h_1 are given by

$$a_1 = 0.0 \tag{75}$$

and

$$h_1 = \mathrm{HP}(h_0, a_1, a_0) \; . \tag{76}$$

The range of the ray after the trace step, x_1, is given by

$$x_1 = \mathrm{RP}(x_0, a_1 - a_0) \; . \tag{77}$$

If a_1 is less than or equal 0.0 and if h_1 is less than or equal $htdum(j_l)$, then

$$\begin{aligned}
h_1 &= htdum(j_l) \\
a_1 &= -\mathrm{SQRT}\left(\mathrm{RADA1}\{a_0, h_1 - h_0\}\right) \\
x_1 &= \mathrm{RP}(x_0, a_1 - a_0) \\
j_l &= j_l - 1 \\
j_l &= 1 \qquad if \;\; (j_l = 0)
\end{aligned} \quad . \tag{78}$$

Else if a_1 is greater to or equal to 0.0 and if h_1 is greater to or equal to $htdum(j_l + 1)$, then

$$
\begin{aligned}
h_1 &= htdum(j_l + 1) \\
a_1 &= \mathrm{SQRT}\left(\mathrm{RADA1}\{a_0, h_1 - h_0\}\right) \\
x_1 &= \mathrm{RP}\left(x_0, a_1 - a_0\right) \\
j_l &= j_l + 1 \\
j_l &= lvlep \qquad if\ (j_l > lvlep)
\end{aligned}
\tag{79}
$$

where *lvlep* is the number of height/refractivity levels in the profile. If h_1 is greater than z_{lim}, then

$$
\begin{aligned}
h_1 &= z_{lim} \\
a_1 &= \mathrm{SQRT}\left(\mathrm{RADA1}\{a_0, h_1 - h_0\}\right) \ . \\
x_1 &= \mathrm{RP}\left(x_0, a_1 - a_0\right)
\end{aligned}
\tag{80}
$$

Finally, in preparation for the next trace step, the parameters a_0, h_0, and x_0 are set equal to a_1, h_1, and x_1, respectively. Further, if the reflected ray hits the ground (i.e., h_0 is less than or equal to 10^{-4}); then a_0 is set to the negative of a_0, and x_{PE} is set to x_0.

Prior to end of the inner DO WHILE statement, several tests are made to determine if the DO WHILE should be terminated. First, if a_0 is greater than $\pi/2$ (i.e., the ray is vertical), then this SU is exited. The maximum of the local angle along the ray a_{mxcurl} is set to the maximum of the current value of a_{mxcurl} or a_0. If h_0 is greater than or equal to z_{lim} and if a_0 is greater than 0.0, *loop* is set to '.false.'. Finally, if x_0 is greater than x_{lim}, then *loop* is false.

At this point the inner DO WHILE loop begins the processing of the next range step if *loop* is true.

If the ray traced does not reach z_{lim} and is not reflected within x_{lim}, then the initial launch angle is increased by 1 milliradian and the ray trace is repeated in the outer DO WHILE loop. The precise test used is given by

$$
i_{set} = 1 \qquad if\ \left(\{r_0 \le x_{lim}\}\ and\ \{x_{PE} > 0.0\}\right) \ . \tag{81}
$$

If this criteria is met, then it is necessary to make sure the local maximum angle, a_{mzcur}, is just within the user specified angle, p_{angle} (if p_{angle} is non-zero). First, for the terrain case, if p_{angle} is greater than 10^{-6}; then i_{set} is set initially to 1. If a_{mzcur} is greater than Θ_{max}, i_{set} is set to 0. If a_s is less than or equal to the angle above which no rays are trapped, a_{crit} plus 10^{-3}, then i_{set} is set to 1 (i.e., the launch angle is not allowed to be less than a_{crit}). If p_{angle} is less than or equal to 10^{-6}, then

(1) if $(x_o \le x_{lim})$ and $(h_o \ge z_{lim})$ i_{set} is set to 1 and

(2) if i_{set} equals 1 then

$$\Theta_{max} = \mathrm{AMAX1}\left(\mathrm{ABS}\{a_s, a_{mxcur}\}\right) \quad . \tag{82}$$

Second, for the smooth earth case, if p_{angle} is greater than 10^{-6} and if i_{set} equals one; then a temporary variable a is determined by

$$a = \mathrm{AMAX1}\left(\mathrm{ABS}\{a_s\}, a_{mxcur}\right) \quad . \tag{83}$$

If a is less than p_{angle}, i_{set} is set to zero. If a is greater or equal to p_{angle} and if a_{sl} is not equal to zero, then

$$\begin{aligned} a_s &= a_{sl} \\ a_{mxcur} &= a_{mxcurl} \end{aligned} \quad . \tag{84}$$

Just as a safeguard, the absolute maximum launch angle is set to fifteen degrees. That is, if a_s is less than or equal to $-15°$, then i_{set} is set to 1, a_s is set to $-15°$ in radians, and a_{mzcur} is set to $15°$ in radians.

Before the end of the outer DO WHILE loop, a_{sl} is set to a_s and a_{mxcurl} is set to a_{mxcurl}.

After the outer DO WHILE loop, a test is made to determine if the case is smooth earth. If it is then,

$$\Theta_{max} = \mathrm{AMAX1}\left(\mathrm{ABS}\{a_s\}, a_{mxcur}\right) \quad . \tag{85}$$

Plus Θ_{launch} is set to the absolute value of a_s.

Table 5-7 and Table 5-8 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the TRACEA SU.

Table 5-7 TRACEA SU Input Data Element Requirements

Name	Description	Units	Source
a_{crit}	Critical angle, angle above which no rays are trapped for ducting environment	radians	CALLING SU
ant_{ref}	Transmitting antenna height relative to the reference height y_{minter}	radians	CALLING SU
$\partial M(j_l)/\partial h$	Gradient of first profile at j_l	M-units /meter	CALLING SU
fter	Logical flag representing terrain	N/A	CALLING SU
htdum	Dummy array containing height values for current (horizontally interpolated) profile	meters	CALLING SU
j_{ls}	Index of the refractivity array at which the antenna height is located	N/A	CALLING SU
lvlep	Number of height/refractivity levels in profile	N/A	CALLING SU
mxlvls	Maximum number of height/M-unit levels	N/A	TPEM.INC
mxter	Maximum number of height/range points allowed for terrain profile	N/A	TPEM.INC
p_{angle}	User specified propagation angle	radians	CALLING SU
slp	Slope of each segment of terrain	meters/ meter	calling su
Θ_{max}	Maximum propagation angle in PE calculations	radians	CALLING SU
tr	Terrain structure for external terrain data elements	N/A	TPEM.INC
tr.terx	Range points of terrain profile	meters	TPEM.INC
tr.tery	Height points of the terrain profile	meters	TPEM.INC
x_{lim}	90% of the maximum range, x_{max}, used for ray tracing	meters	CALLING SU
x_{PE}	Range at which valid loss values will begin to be calculated	meters	CALLING SU
z_{lim}	Height limit for ray trace	meters	CALLING SU

Table 5-8 TRACEA SU Output Data Element Requirements

Name	Description	Units	Source
Θ_{launch}	Launch angle	radians	TRACEA SU
Θ_{max}	Maximum propagation angle in PE calculations	radians	TRACEA SU

5.1.4 Dielectric Initialization (DIEINIT) SU

The purpose of the DIEINIT SU is to determine the conductivity and relative permittivity as a function of frequency in MHz based on general ground composition types.

It supports the ground types: salt water, fresh water, wet ground, medium dry ground, very dry ground, and user defined. For all ground types other than user defined, the permittivity and conductivity are calculated as functions of frequency from curve fits to the permittivity and conductivity graphs shown in the Recommendations and Reports of the International Radio Consultative Committee (CCIR, 1986). For $tr.i_{gr}$ input ground types cases *tr.igrnd(i)*, the permittivity and conductivity are determined as follows..

For salt water (case 0), the relative permittivity is given by 70 for frequencies $f_{MHz} \le 2253.5895$ MHz; and the conductivity is given by 5.0 S/m for $f_{MHz} \le 1106.207$. For $f_{MHz} > 2253.5895$ MHz, the relative permittivity ε_r is given by

$$\varepsilon_r = \begin{bmatrix} 1.4114535{*}10^{-2} - 5.2122497{*}10^{-8} f_{MHz} + 5.8547829 * 10^{-11} f_{MHz}^2 \\ -7.6717423 * 10^{-16} f_{MHz}^3 + 2.9856318 * 10^{-21} f_{MHz}^4 \end{bmatrix}^{-1} . \quad (86)$$

For $f_{MHz} > 1106.207$ MHz, the conductivity σ in S/m is given by

$$\sigma = \frac{3.8586749 + 9.1253873 * 10^{-4} f_{MHz} + 1.530992 * 10^{-8} f_{MHz}^2}{1.-2.1179295 * 10^{-5} f_{MHz} + 6.5727504 * 10^{-10} f_{MHz}^2 - 1.9647664 * 10^{-15} f_{MHz}^3} . \quad (87)$$

For fresh water (case 1), the relative permittivity ε_r is given by 80 for frequencies $f_{MHz} \le 6165.776$ MHz. For higher frequencies, ε_r is given by

$$\varepsilon_r = \frac{79.027635 - 3.5486605 * 10^{-4} f_{MHz} + 8.210184 * 10^{-9} f_{MHz}^2}{1.-2.2083308 * 10^{-5} f_{MHz} + 2.7067836 * 10^{-9} f_{MHz}^2 - 1.0007669 * 10^{-14} f_{MHz}^3} . \quad (88)$$

For $f_{MHZ} > 5776.157$ MHz, the conductivity σ in S/m is given by

$$\sigma = \left(\frac{-0.65750351 + 6.6113198 * 10^{-4} f_{MHz} + 1.4876952 * 10^{-9} f_{MHz}^2}{1.+5.5620223 * 10^{-5} f_{MHz} + 3.0140816 * 10^{-10} f_{MHz}^2} \right)^2 . \quad (89)$$

At $f_{MHZ} \leq 5776.157$ MHz, the conductivity σ in S/m is given by

$$\sigma = \left(\frac{201.97103 + 1.2197967 * 10^{-2} f_{MHz} - 1.728776 * 10^{-6} f_{MHz}^2}{1.-2.5539582 * 10^{-3} f_{MHz} - 3.7853169 * 10^{5} f_{MHz}^2} \right)^{-1} . \tag{90}$$

For wet ground (case 2), the relative permittivity ε_r is given by 30 for frequencies $f_{MHz} \leq 1312.054$ MHz. For frequencies $1312.054 < f_{MHz} < 4228.11$ MHz, the relative permittivity ε_r is given by

$$\varepsilon_r = \sqrt{\frac{857.94335 + 5.5275278 * 10^{-2} f_{MHz}}{1.-8.9983662 * 10^{-5} f_{MHz} + 8.8247139 * 10^{-8} f_{MHz}^2}} . \tag{91}$$

For frequencies $f_{MHz} \geq 4228.11$ MHz, the relative permittivity ε_r is given by

$$\varepsilon_r = \sqrt{\frac{915.31026 - 4.0348211 * 10^{-3} f_{MHz} + 7.4342897 * 10^{-7} f_{MHz}^2}{1.-9.4530022 * 10^{-6} f_{MHz} + 4.892281 * 10^{-8} f_{MHz}^2}} . \tag{92}$$

For frequencies $f_{MHz} > 15454.4$ MHz, the conductivity σ in S/m for wet ground is given by

$$\begin{aligned} \sigma = {} & 0.8756665 + 4.7236085 * 10^{-5} f_{MHz} + 2.6051966 * 10^{-8} f_{MHz}^2 \\ & - 9.235936 * 10^{-13} f_{MHz}^3 + 1.4560078 * 10^{-17} f_{MHz}^4 \\ & - 1.1129348 * 10^{-22} f_{MHz}^5 + 3.3253339 * 10^{-28} f_{MHz}^6 \end{aligned} . \tag{93}$$

For frequencies $f_{MHz} \leq 15454.4$ MHz, the conductivity σ in S/m for wet ground is given by

$$\begin{aligned} \sigma = {} & 5.5990969 * 10^{-3} + 8.7798277 * 10^{-5} f_{MHz} + 6.2451017 * 10^{-8} f_{MHz}^2 \\ & - 7.1317207 * 10^{-12} f_{MHz}^3 + 4.2515914 * 10^{-16} f_{MHz}^4 \\ & - 1.240806 * 10^{-20} f_{MHz}^5 + 1.3854354 * 10^{-25} f_{MHz}^6 \end{aligned} . \tag{94}$$

For medium dry ground (case 3), the relative permittivity ε_r is given by 15 for frequencies $f_{MHz} \leq 4841.945$ MHz. For frequencies $f_{MHz} > 4841.945$ MHz, the relative permittivity ε_r is given by

$$\varepsilon_r = \sqrt{\frac{215.87521 - 2.6151055 * 10^{-3} f_{MHz} + 1.9484482 * 10^{-7} f_{MHz}^2}{1. - 7.6649237 * 10^{-5} f_{MHz} + 1.2565999 * 10^{-8} f_{MHz}^2}} \quad . \tag{95}$$

At frequencies $f_{MHz} \le 4946.751$ MHz for medium dry ground, the conductivity σ in S/m is given by

$$\begin{aligned} \sigma = (&2.4625032 * 10^{-2} + 1.8254018 * 10^{-4} f_{MHz} - 2.664754 * 10^{-8} f_{MHz}^2 \\ &+ 7.6508732 * 10^{-12} f_{MHz}^3 - 7.4193268 * 10^{-16} f_{MHz}^4)^2 \end{aligned} \quad . \tag{96}$$

At frequencies $f_{MHz} > 4946.751$ MHz for medium dry ground, the conductivity σ in S/m is given by

$$\begin{aligned} \sigma = (&0.17381269 + 1.2655183 * 10^{-4} f_{MHz} - 1.6790756 * 10^{-9} f_{MHz}^2 \\ &+ 1.1037608 * 10^{-14} f_{MHz}^3 - 2.9223433 * 10^{-20} f_{MHz}^4)^2 \end{aligned} \quad . \tag{97}$$

For very dry ground (case 4), the relative permittivity ε_r is given by 3 and the conductivity σ in S/m is 0.0001 for frequencies $f_{MHz} < 590.8924$ MHz. For frequencies $590.8924 \le f_{MHz} \le 7131.933$ MHz, the conductivity σ in S/m is given by

$$\begin{aligned} \sigma = &2.2953743 * 10^{-4} - 8.1212741 * 10^{-7} f_{MHz} + 1.8045461 * 10^{-9} f_{MHz}^2 \\ &- 1.960677 * 10^{-12} f_{MHz}^3 + 1.256959 * 10^{-15} f_{MHz}^4 - 4.46811 * 10^{-19} f_{MHz}^5 \\ &+ 9.4623158 * 10^{-23} f_{MHz}^6 - 1.1787443 * 10^{-26} f_{MHz}^7 + 7.9254217 * 10^{-31} f_{MHz}^8 \\ &- 2.2088286 * 10^{-35} f_{MHz}^9 \end{aligned} \tag{98}$$

For frequencies $f_{MHz} > 7131.933$ MHz, the conductivity σ in S/m is given by

$$\begin{aligned} \sigma = (&-4.9560275 * 10^{-2} + 2.9876572 * 10^{-5} f_{MHz} - 3.0561848 * 10^{-10} f_{MHz}^2 \\ &+ 1.1131828 * 10^{-15} f_{MHz}^3)^2 \end{aligned} \quad . \tag{99}$$

For the user defined ground type (case 5), the relative permittivity ε_r and the conductivity σ in S/m are set equal to the input values *tr.dielect(1,i)* and *tr.dielect(2,i)*, respectively.

Finally, the dielectric constants at a range of $1*10^7$ km, are set to the values last determined for the $tr.i_{gr}{}^{\text{th}}$ case, $tr.i_{gr}$ is increased by 1, and the range at which these dielectric constants apply is stored in the $(tr.i_{gr}+1)^{\text{th}}$ position of *tr.rgrnd*.

Table 5-9 and Table 5-10 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the DIEINIT SU.

Table 5-9 DIEINIT SU Input Data Element Requirements

Name	Description	Units	Source
f_{MHz}	Frequency	MHz	TPRM.INC
sv	System structure for external system data elements	N/A	TPEM.INC
tr	Terrain structure for external terrain data elements	N/A	TPEM.INC SU
tr.dielec	2-dimensional array containing the relative permittivity and conductivity for user defined terrain	$^{\alpha}$	TPEM.INC
$tr.i_{gr}$	Number of different ground types specified	N/A	TPEM.INC
tr.igrnd	Type of ground composition for given terrain profile	N/A	TPEM.INC

$^{\alpha}$ conductivity has units of S/m.

Table 5-10 DIEINIT SU Output Data Element Requirements

Name	Description	Units	Source
$tr.i_{gr}$	Number of different ground types specified	N/A	DIEINIT SU
tr.dielec	2-dimensional array containing the relative permittivity and conductivity$^{\alpha}$ for user defined terrain	N/A	DIEINIT SU
tr.rgrnd	Ranges at which the ground types apply	meters	DIEINIT SU

$^{\alpha}$ conductivity has units of S/m.

5.1.5 Get FFT Size (GETFFTSZ) SU

The purpose of the GETFFTSZ SU is to determine the required transform size, n, based on the maximum PE propagation angle and the specified maximum output height. If the transform size required is greater than the maximum allowed, then the maximum PE height calculation volume is calculated based on the maximum allowable transform size. Propagation loss is provided only up to the maximum PE calculation height or the specified maximum output height, whichever is smaller.

First, some constants used within the PE solution are initialized. The bin width in z space is given in terms of the PE calculation maximum propagation angle, Θ_{max}, and wavelength, λ, by

$$\Delta z_{PE} = \frac{\lambda}{2\, \mathrm{SIN}\left(\Theta_{max}\right)} \ . \tag{100}$$

The total number of vertical points for which a transformation will be computed, n_{fft}, is determined as $2^{ln_{fft}}$. This term is also referred to as the FFT size. The minimum size for the parameter ln_{fft} is set to 9 for smooth surfaces and 10 for terrain cases. The total height of the PE calculation domain, z_{max}, is initialized as

$$z_{max} = n_{fft}\, \Delta z_{PE} \ . \tag{101}$$

For computational efficiency reasons, an artificial upper boundary must be established for the PE solution. To prevent upward propagating energy from being "reflected" downward from this boundary and contaminating the PE solution, the upper one quarter of the PE solution field strength is attenuated or "filtered" to insure that the field strength just below the upper boundary is reduced to zero. An iteration using equations 102-104 is repeated until height z_{limit}, the maximum height region where the PE solution is valid, satisfies $\frac{3z_{max}}{4} \geq z_{limit}$. The parameters ln_{fft}, n_{fft}, and z_{max} are given by

$$ln_{fft} = ln_{fft} + 1 \tag{102}$$

$$n_{fft} = 2^{\ln_{fft}} \tag{103}$$

$$z_{max} = n_{fft} \Delta z_{PE} \quad . \tag{104}$$

If the transform size needed is too large (i.e., $ln_{fft} > 14$), then ln_{fft}, n_{fft}, z_{max}, and z_{limit} are set accordingly as follows

$$ln_{fft} = mxnfft \tag{105}$$

$$n_{fft} = 2^{ln_{fft}} \tag{106}$$

$$z_{\max} = n_{fft} \Delta z_{PE} \tag{107}$$

$$z_{limit} = \frac{3}{4} z_{fft} \quad . \tag{108}$$

Table 5-11 and Table 5-12 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element respectively of the GETFFTSZ SU.

Table 5-11 GETFFTSZ SU Input Data Element Requirements

Name	Description	Units	Source
fter	Logical flag indicating a terrain case is being performed	N/A	PEINIT CSC
λ	Wave length	meters	PEINIT CSC
mxnfft	Maximum power of 2 for transform size	N/A	FFTSIZ.INC
Θ_{max}	Maximum propagation angle in PE calculations	radians	PEINIT CSC
z_{limit}	Maximum height region where PE solution is valid	meters	Calling SU

Table 5-12 GETFFTSZ SU Output Data Element Requirements

Name	Description	Units	Source
Δz_{PE}	Bin width in z space	meters	GETFFTSZ SU
ln_{fft}	Power of 2 transform size, i.e. $n_{fft} = 2**ln_{fft}$	N/A	GETFFTSZ SU
n_{fft}	Transform size	N/A	GETFFTSZ SU
z_{limit}	Maximum height region where PE solution is valid	meters	GETFFTSZ SU
z_{max}	Total height of the FFT/PE calculation domain	meters	GETFFTSZ SU

5.1.6 Starter Field Initialization (XYINIT) SU

The purpose of the XYINIT SU is to calculate the complex PE solution at range zero.

Upon entering this SU, several constant terms which will be employed over the entire PE mesh are calculated. The PE mesh is defined by the number of points in the mesh, n_{fft}, and by the mesh size Δp. The constant terms include : (1) the angle difference between mesh points in p-space $\Delta\Theta$; (2) a height-gain value at the source (transmitter) ant_{k_o}; (3) the normalization factor s_{gain} used in the determination of the complex array containing the field U; (4) the default value of the complex reflection coefficient *refcoef* and (5) the complex index of refraction for the vertical polarization case. The complex index of refraction for vertical polarization (*sv.polar* = 'V') is obtained from reference to the GETALN SU for the terrain structure *tr*. The parameter used as the result of this reference is the complex refractive index squared R_{ng2}. The complex reflection coefficient *refcoef* is given by

$$refcoef = \mathrm{CMPLX}\left(-1.,0.\right) \quad . \tag{109}$$

The normalization factor s_{gain} is given by

$$s_{gain} = \frac{\mathrm{SQRT}\left(\lambda\right)}{z_{max}} \quad . \tag{110}$$

The angle difference between mesh points in p-space, $\Delta\Theta$, is given by

$$\Delta\Theta = \frac{\Delta p}{k_o} \tag{111}$$

where k_o is the free space wave number. The height-gain value at the source (transmitter) ant_{k_o} is given by

$$ant_{k_o} = k_o \; h_{transmitter} \tag{112}$$

where $h_{transmitter}$ is the transmitting antenna height above the local ground in meters.

For each point in the PE p-space mesh (i.e., i=0, n_{fft}); the following steps are performed. The direct-path ray elevation angle p_k is determined from

$$p_k = \mathrm{FLOAT}\left(i\right) \Delta\Theta \ . \tag{113}$$

The antenna pattern ANTPAT SU is referenced with the antenna pattern type, i_{pat}, and the elevation angle p_k (negative value for the surface-reflected ray) to obtain antenna pattern factors, fac_D and fac_R, for both a direct-path ray and a surface-reflected ray, respectively. For vertical polarization (*sv.polar* = 'V') the complex surface reflection coefficient is determined with the grazing angle equal to the negative of the reflected ray angle, or in this case, the direct-path ray elevation angle. For the complex refractive index squared, R_{ng2}, previously obtained from the reference to GELATIN SU above, the reflection coefficient *refcoef* is given by

$$refcoef = \frac{R_{ng2}\,\mathrm{SIN}\left(p_k\right) - \mathrm{CSQRT}\left(R_{ng2} - \mathrm{COS}^2\left\{p_k\right\}\right)}{R_{ng2}\,\mathrm{SIN}\left(p_k\right) + \mathrm{CSQRT}\left(R_{ng2} - \mathrm{COS}^2\left\{p_k\right\}\right)} \ . \tag{114}$$

The complex portions of the PE solution U are determined from the antenna pattern factors, reflection coefficient, elevation angle, and normalization factor from

$$U(i) = s_{gain}\left(fac_D \ D_{term} + refcoef \ fac_R \ R_{term}\right) \tag{115}$$

where the field, R_{term}, due to an image point source at the height $-h_{transmitter}$ is given by

$$R_{term} = \mathrm{CMPLX}\left(\mathrm{COS}\left\{p_k \, ant_{k_o}\right\}, \mathrm{SIN}\left\{p_k \, ant_{k_o}\right\}\right) \tag{116}$$

and where the field, D_{term}, due to a real point source at the height $h_{transmitter}$ is given by

$$D_{term} = \mathrm{CONJG}\left(R_{term}\right) \ . \tag{117}$$

Finally, the upper ¼ of the field values are filtered. A cosine -tapered (Tukey) filter array *FILT* is used for this purpose. For $\left(i = n_{3/4} \text{ to } n_{nfft}\right)$, the field $U(i)$ is given by

$$U(i) = U(i)\,FILT\left(i - n_{3/4}\right) \tag{118}$$

where $n_{3/4}$ is equal to ¾ of n_{fft}.

Table 5-13 and Table 5-14 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the XYINIT SU.

Table 5-13 XYINIT SU Input Data Element Requirements

Name	Description	Units	Source
Δp	Mesh size in angle- (or p-) space	radians	PEINIT CSC
FILT	Cosine-tapered (Tukey) filter array		PEINIT CSC SU
$h_{transmitter}$	Transmitting antenna height above the local ground	meters	TPEM.INC
i_{pat}	Type of antenna pattern desired	N/A	TPEM.INC
k_o	Free-space wave number $= 2\pi / \lambda$	meters^{-1}	PEINIT CSC
λ	Wavelength	meters	PEINIT CSC
maxn4	*maxpts* divided by 4; specifies the length of the filter array	N/A	TPEM.INC
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	FFTSIZ.INC
n_{fft}	Transform size	N/A	GETFFTSZ SU
$n_{3/4}$	¾ of n_{fft}	N/A	PEINIT CSC SU
R_{ng2}	Complex refractive index squared	N/A	GETALN SU
sv	System structure for external system data elements	N/A	TPEM.INC
sv.polar	Character string indicating polarization	N/A	TPEM.INC
tr	Terrain structure for external terrain data elements	N/A	TPEM.INC
z_{max}	Total height of the FFT/PE calculation domain	meters	GETTFFTSZ SU

Table 5-14 XYINIT SU Output Data Element Requirements

Name	Description	Units	Source
U	Transform of complex field		XYINIT SU

5.1.7 Fast-Fourier Transform (FFT) SU

The purpose of the FFT SU is to separate the real and imaginary components of the complex PE field, respectively, into two real arrays and then to reference the SINFFT SU which transforms each portion of the PE solution.

For a transform size, n_{fft}, the real and imaginary parts of the complex PE field array U, respectively, are found for the index i from 0 to n_{fft}.

$$X(i) = \mathrm{REAL}\left(U\{i\} \right) \tag{119}$$

and

$$Y(i) = \mathrm{IMAG}\left(U\{i\} \right) \tag{120}$$

The transform size, n_{fft}, is constrained to be less than or equal to 2^{14} . The SINFFT SU is referenced in turn for X and Y along with ln_{fft}, the power of the transform size to the base 2 $\left(n_{fft} = 2^{ln_{fft}} \right)$. The real and imaginary parts of the resulting transform arrays, X and Y, are then converted to the imaginary array U for i equal 0 to n_{fft} by

$$U(i) = \mathrm{CMPLX}\left(X\{i\}, Y\{i\} \right) \ . \tag{121}$$

Table 5-15 and Table 5-16 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the FFT SU.

Table 5-15 FFT SU Input Data Element Requirements

Name	Description	Units	Source
ln_{fft}	Power of 2 transform size, i.e. n=2**ln	N/A	GETFFTSZ SU
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	FFTSIZ.INC
n_{fft}	Transform size	N/A	GETFFTSZ SU
U	Complex field to be transformed	$\mu V / m$	Calling SU

Table 5-16 FFT SU Output Data Element Requirements

Name	Description	Units	Source
U	Transform of complex field	$\mu V / m$	FFT SU

5.1.8 Sine Fast-Fourier Transform (SINFFT) SU

A function with a common period, such as a solution to the wave equation, may be represented by a series consisting of sines and cosines. This representation is known as a Fourier series. An analytical transformation of the function, known as a Fourier transform, may be used to obtain a solution for the function.

The solution to the PE approximation to Maxwell's wave equation is obtained by using such a Fourier transformation function. The TPEM CSCI uses only the real-valued sine transformation in which the real and imaginary parts of the PE equation are transformed separately. The Fourier transformation provided with the TPEM CSCI is described by Bergland (1969) and Cooley (1970). The FORTRAN source code is listed in APPENDIX A.

Other sine fast Fourier transform (FFT) routines are available in the commercial market, and such a sine FFT may already be available within another TESS CSCI. The selection of which FFT ultimately used by the TPEM CSCI is left to the application designer as every sine FFT will have hardware and/or software performance impacts. For this reason, it is beyond the scope of this document to describe the numerical implementation of the FFT algorithm.

Table 5-17 and Table 5-18 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the SINFFT SU.

Table 5-17 SINFFT Input Data Element Requirements

Name	Description	Units	Source
n_{fft}	Transform size	N/A	GETFFTSZ SU
U	Complex field to be transformed dimensioned $2^{n_{fft}}$ in calling SU		FFT SU

Table 5-18 SINFFT Output Data Element Requirements

Name	Description	Units	Source
U	Sine transform of complex field		SINFFT SU

5.1.9 Trace Launch Angle (TRACEH) SU

The purpose of the TRACEH SU is to perform a ray trace for a single ray and store all heights traced to each output range step. The initial launch angle is the negative of the input launch angle Θ_{launch} for a smooth surface. For the terrain case, the initial launch angle is Θ_{launch}. Upon reflection the heights of this ray at each output range point x_o are then stored in array $y_{\lim}$ for subsequent output of loss values in array *mloss*. This is done so that only loss values that fall within the valid PE solution region are output or passed back in *mloss*.

First, the following internal functions are defined.

$$\mathrm{RADA1}\left(a, b\right) = a^2 + 2 \; grad \; b \tag{122}$$

$$\mathrm{RP}\left(a, b\right) = a + \frac{a}{grad} \tag{123}$$

$$\mathrm{AP}\left(a, b\right) = a + b \; grad \tag{124}$$

$$\mathrm{HP}\left(a, b, c\right) = a + \frac{b^2 - c^2}{2 \; grad} \tag{125}$$

Initially, several constants are set. The internal variable for the starting launch angle, a_0, is set to the negative of the input launch angle, Θ_{launch}. If this case is a terrain case, then a_0 is set to Θ_{launch}. The height of the ray before the trace step is set equal to ant_{ref}, the transmitting antenna height relative to the reference height y_{minter}. The index of the current refractivity layer for which the ray is being traced through, j_l, is set equal to j_{ls}. the index of the refractivity array for which the antenna height is located. The current output range at which to store the height of traced ray in y_{lim}, x_o, is set equal to Δx_{out}, the output range step. The variables i_{hu}, the range index at which the traced ray has reached the maximum calculation height; x_0, the range of the ray before the trace step; and x_{PE}, the range at which valid loss values will begin to be calculated, are all set to zero.

A FORTRAN DO loop is used to trace the ray output points. The index for the loop *i* goes from 1 through *nvrout*.

Within this loop a FORTRAN DO WHILE loop is used to trace the ray until it reaches the output range point x_o (i.e., x_0 is greater or equal to x_o). The computation within this nested loop is as follows. First, the range of the ray after the trace step, x_1, is set equal to x_o. Then, the variable *grad*, the gradient of the current refractivity layer, is set equal to $\partial M(j_l)/\partial h$, the gradient of the first profile at index j_l. The value of the angle after the trace step, a_1, is found from

$$a_1 = \mathrm{HP}\left(a_0, x_1 - x_0\right) \ . \tag{126}$$

If $\mathrm{SIGN}\left(1.0, a_0\right)$ is not equal to $\mathrm{SIGN}\left(1.0, a_1\right)$, then a_0 and x_1, respectively, are given by

$$a_1 = 0.0 \tag{127}$$

and

$$x_1 = \mathrm{RP}\left(x_0, a_1 - a_0\right) \ . \tag{128}$$

The height of the ray after the trace step, h_1, is given by

$$h_1 = \mathrm{HP}\left(h_0, a_1, a_0\right) \ . \tag{129}$$

If a_1 is less than or equal 0.0 and if h_1 is less than or equal $htdum\left(j_l\right)$, then

$$\begin{aligned} h_1 &= htdum\left(j_l\right) \\ a_1 &= -\mathrm{SQRT}\left(\mathrm{RADA1}\left\{a_0,\ h_1 - h_0\right\}\right) \\ x_1 &= \mathrm{RP}\left(x_0, a_1 - a_0\right) \\ j_l &= j_l - 1 \\ j_l &= 1 \qquad \text{if } \left(j_l = 0\right) \end{aligned} \quad . \tag{130}$$

Else if a_1 is greater than or equal to 0.0 and if h_1 is greater than or equal to $htdum\left(j_l + 1\right)$, then

$$
\begin{aligned}
h_1 &= htdum\left(j_l + 1\right) \\
a_1 &= \mathrm{SQRT}\left(\mathrm{RADA1}\left\{a_0, h_1 - h_0\right\}\right) \\
x_1 &= \mathrm{RP}\left(x_0, a_1 - a_0\right) \\
j_l &= j_l + 1 \\
j_l &= lvlep \qquad \text{if } \left(\mathrm{j}_1 > lvlep\right)
\end{aligned}
\tag{131}
$$

where *lvlep* is the number of height/refractivity levels in the profile. If x_1 is greater than x_0, then the values of x_1, a_1, and h_1 are given by

$$
\begin{aligned}
x_1 &= x_o \\
a_1 &= \mathrm{AP}\left(a_0, x_1 - x_0\right) \; . \\
h_1 &= \mathrm{HP}\left(h_0, a_1, a_0\right)
\end{aligned}
\tag{132}
$$

The values of x_0, a_0, and h_0 are then set equal to the respective values of x_1, a_1, and h_1 so determined. Further, if the reflected ray hits the ground (i.e., h_0 is less than or equal to 10^{-4}); then a_0 is set to the negative of a_0, and x_{PE} is set to x_0. If the ray traced reaches the height limit for ray tracing, z_{lim}, then all heights for this ray for subsequent output range points will also be z_{lim}. In this case the index i_{hu} is set equal to *i* and TRACEH SU is exited ending the DO WHILE loop.

Within the DO loop itself, final processing of the loop occurs. First, a test is made to determine whether i_{hu} is greater than zero. If it is, then the TRACEH SU is exited. If it is not, then a test is made to determine if a_0 is less than zero. If it is then $y_{lim}(i)$ is set equal to zero. Further, if a_0 is greater or equal to zero, then $y_{lim}(i)$ is set equal to h_0. Finally, x_o is incremented by the value of Δx_{out}.

After the DO loop has been exited, a test is made to determine if i_{hu} is greater than zero. If it is, then the elements of y_{lim} are set equal to *htlim* for values of the index *i* running from i_{hu} to *nvrout.*

Table 5-19 and Table 5-20 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the TRACEH SU.

Table 5-19 TRACEH SU Input Data Element Requirements

Name	Description	Units	Source
ant_{ref}	Transmitting antenna height relative to the reference height y_{minter}	meters	Calling SU
$\partial M(\ )/\partial h$	Gradients of first profile	M-units/ meter	Calling SU
Δx_{out}	Output range step	meters	Calling SU
fter	Logical flag indicating if performing terrain case	N/A	Calling SU
htdum	Dummy array containing height values for current (horizontally interpolated) profile	meters	Calling SU
htlim	Maximum desired calculation height with respect to y_{minter}	meters	CSC PEINIT
j_{ls}	Index of the refractivity array at which the antenna height is located	N/A	Calling SU
lvlep	Number of height/refractivity levels in profile	N/A	Calling SU
mxlvls	Maximum number of height/M-unit levels	N/A	TPEM.INC
mxrout	Maximum number of output range points	N/A	TPEM.INC
nvrout	Number of output range points	N/A	TPEM.INC
Θ_{launch}	Launch angle	radians	Calling SU
x_{lim}	90% of the maximum range, x_{max}	meters	CSC PEINIT
z_{lim}	Height limit for ray trace	meters	CSC PEINIT

Table 5-20 TRACEH SU Output Data Element Requirements

Name	Description	Units	Source
x_{PE}	Range at which valid loss values will begin to be calculated	meters	TRACEA SU
y_{lim}	Height at each output range at which the last valid loss value exists	meters	TRACEH SU

5.1.10 Free-Space Propagator Phase Term (PHASE1) SU

The purpose of the PHASE1 SU is to initialize the free-space propagator array for subsequent use in the PESTEP CSC. The propagator term is computed at each PE angle, or p-space, mesh point using the wide-angle propagator. Finally, a filter, or attenuation function (frequently called "window"), is applied to the upper one-quarter of the array corresponding to the highest one-quarter of the maximum propagation angle.

For vales of the index i running from 0 to n_{fft}, the complex array containing the free-space propagator terms $frsp(i)$ is given by

$$frsp(i) = f_{norm}\ \mathrm{CMPLX}\left(\mathrm{COS}\left\{\Delta x_{k_0}\,[1.0 - c_{ak}\,]\right\}, -\mathrm{SIN}\left\{\Delta x_{k_0}\,[1.0 - c_{ak}\,]\right\}\right) \tag{133}$$

where Δx_{k_0} and c_{ak}, a double precision variable, are given, respectively, by

$$\Delta x_{k_0} \;=\; k_0\,\Delta x_{PE} \tag{134}$$

and

$$c_{ak} \;=\; \mathrm{SQRT}\left(1.0 - \mathrm{AMIN1}\left\{1.0, [\,cnst * \mathrm{FLOAT}\,(\,i\,)]^2\,\right\}\right)\ . \tag{135}$$

The upper ¼ of the free-space propagator term, *frsp*, is filtered by a cosine-tapered (Tukey) filter array, *FILT*. For values of i running from $n_{3/4}$ to n_{fft}, $frsp(i)$ is given by

$$frsp\left(\,i\,\right) \;=\; FILT\left(\,i - n_{3/4}\right)\ frsp\left(\,i\,\right)\ . \tag{136}$$

Table 5-21 and Table 5-22 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the PHASE1 SU.

Table 5-21 PHASE1 SU Input Data Element Requirements

Name	Description	Units	Source
cnst	Constant equals $\Delta p / k_o$	radians * meters	PEINIT CSC
Δx_{PE}	PE range step	meters	Calling SU
FILT	Cosine-tapered (Tukey) filter array		PEINIT CSC
f_{norm}	Normalization factor	N/A	PEINIT CSC
k_o	Free-space wave number $= 2\pi / \lambda$	meters^{-1}	PEINIT CSC
maxn4	*maxpts* divided by 4; specifies the length of the filter array	N/A	TPEM.INC
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	TPEM.INC
n_{fft}	Transform size	N/A	GETFFTSZ SU
$n_{3/4}$	¾ of n_{fft}	N/A	PEINIT CSC SU

Table 5-22 PHASE1 SU Output Data Element Requirements

Name	Description	Units	Source
frsp	Complex free space propagator term array	N/A	PHASE1 SU

5.1.11 Environmental Propagator Phase Term (PHASE2) SU

The purpose of the PHASE2 SU is to calculate the environmental phase term for an interpolated environment profile. This environmental phase term is computed at each PE height, or z-space, mesh point. Finally, a filter, or attenuation function (frequently called "window"), is applied to the upper one-quarter of the array corresponding to the highest one-quarter of the calculation height domain.

For values of the index i running from 0 to n_{fft}, the complex array containing the refractivity profile array interpolated every Δz_{PE} in height is given by

$$envpr(i) = \mathrm{CMPLX}\left(\mathrm{COS}\left\{\Delta x_{PE}\ \ profint[i]\right\}, \mathrm{SIN}\left\{\Delta x_{PE}\ \ profint[i]\right\}\right)\ . \quad (137)$$

The upper ¼ of the *envpr* array is filtered by a cosine-tapered (Tukey) filter array, *FILT*. For values of i running from $n_{3/4}$ to n_{fft}, *envpr*(i) is given by

$$envpr(i) = FILT\left(i - n_{3/4}\right)\ envpr(i)\ . \quad (138)$$

Table 5-23 and Table 5-24 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the PHASE2 SU.

Table 5-23 PHASE2 SU Input Data Elements Requirements

Name	Description	Units	Source
Δx_{PE}	PE range step	meters	Calling SU
FILT	Cosine-tapered (Tukey) filter array		PEINIT CSC
maxn4	*maxpts* divided by 4; specifies the length of the filter array	N/A	TPEM.INC
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	TPEM.INC
n_{fft}	Transform size	N/A	GETFFTSZ SU
$n_{3/4}$	¾ of n_{fft}	N/A	PEINIT CSC SU
profint	Profile interpolated to every Δz_{PE} in height	M-units	Calling SU

Table 5-24 PHASE2 SU Output Data Element Requirements

Name	Description	Units	Source
envpr	Complex refractivity profile array interpolated every Δz_{PE} in height	M-units	PHASE2 SU

5.1.12 Profile Reference (PROFREF) SU

The purpose of the PROFREF SU is to adjust the current refractivity profile so that it is relative to a reference height, y_{ref}. The reference height is initially the minimum height of the terrain profile. Upon subsequent calls from the PESTEP CSC, the refractivity profile is adjusted by the local ground height at each PE range step.

The reference height y_{ref} itself, depending on the value of i_{flag}, can be either y_{minter} or the local ground height above y_{minter}. If i_{flag} is zero, then the profile arrays ref_{ref} and h_{ref} will be relative to y_{minter} and will also be used to initialize *refdum* and *htdum* . If i_{flag} is one, then the profile arrays ref_{ref} and h_{ref} will be referenced to local ground height. The parameter y_{minter} is the reference height for internal calculations in the TPEM CSCI of the complex field *U*. Both the arrays *refdum* and *htdum*, are dummy arrays containing refractivity values and height values, respectively, for the currently interpolated profile.

The determination of the profile arrays ref_{ref} and h_{ref} proceeds as follows. First, the index n_{lvl} is set equal to the input index *lvlep*. Next, a test is made to determine whether the absolute value of the reference height y_{ref} is greater than 10^{-3} (i.e., is y_{ref} greater than approximately zero). If y_{ref} is approximately zero; then, the elements of ref_{ref} are set equal to the corresponding input values of *refdum*, and the elements of h_{ref} are set equal to the corresponding input values of *htdum*.

For the case when y_{ref} is not zero, then the following calculations are made. First, two internal indices, i_{bmsl} and j_s are set equal to zero. Next, the *mxlvls* elements of h_{ref} and ref_{ref} are initialized to zero. Then, y_{ref} is tested to see if it is below sea level (i.e., $y_{ref} < htdum(1)$). If it is, then i_{bmsl} and j_s are set equal to one. If y_{ref} is not below sea level, then the refractivity profile level at which y_{ref} is just above is determined. This test is conducted for values of the index i running from 1 to n_{lvl} -1. The index j_s is set equal to the value of the index i that satisfies both conditions $\left(y_{ref} \le htdum\{i+1\}\right)$ and $\left(y_{ref} > htdum\{i\}\right)$.

If the reference height y_{ref} is not zero and either the conditions $\left(j_s \ne 0\right)$ or $\left(i_{bsml} \equiv 1\right)$ are true; the refractivity at the ground is determined, and the arrays ref_{ref}

and h_{ref} are filled with refractivity profile data where the height zero now refers to the ground reference (i.e., either ground height or y_{minter}). The refractivity at the ground is given by

$$ref_{ref}(1) = redum(j_s) + frac\left(refdum\{j_s+1\} - refdum\{j_s\}\right) \tag{139}$$

where the internal variable *frac* is given by

$$frac = \frac{y_{ref} - htdum(j_s)}{htdum(j_s+1) - htdum(j_s)} \quad . \tag{140}$$

Naturally, the value of h_{ref} at the ground, $h_{ref}(1)$, is set to zero. A test is made to see if $\left(\text{INT}\{frac\} \equiv 1\right)$ is true. If it is, j_s is set equal to j_s plus one. A new level index *newl* is found from $\left(newl = n_{nvl} - j_s + 1\right)$, and a beginning value for the index *k* is found from $\left(k = j_s + 1\right)$. For values of the index j_k running from 2 to *newl,* the arrays ref_{ref} and h_{ref} are evaluated in the order given from *refdum* and *htdum* as follows

$$\begin{aligned} ref_{ref}(j_k) &= refdum(k) \\ h_{ref}(j_k) &= htdum(k) - y_{ref} \quad . \\ k &= k + 1 \end{aligned} \tag{141}$$

The number of levels in the new profile, n_{lvl}, is given by $\left(n_{lvl} = newl\right)$.

Finally, if input index i_{flag} equals zero, then the arrays *refdum* and *htdum* are initialized. Each element of *refdum* is set equal to each element of ref_{ref}, each element of *htdum* is set equal to each element of h_{ref}, and *lvlep* is set equal to n_{lvl}.

Table 5-25 and Table 5-26 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the PROFREF SU.

Table 5-25 PROFREF SU Input Data Element Requirements

Name	Description	Units	Source
htdum	Dummy array containing height values for current (horizontally interpolated) profile	meters	Calling SU
i_{flag}	Index indicating whether the refractivity profile is to be referenced to y_{minter} or to local ground height above y_{minter}	N/A	PEINIT CSC
lvlep	Number of height/refractivity levels in profile	N/A	Calling SU
mxlvls	Maximum number of height/M-unit levels	N/A	TPEM.INC
refdum	Dummy array containing refractivity values for current (horizontally interpolated) profile	M-units	Calling SU
y_{ref}	Reference height at current range step	meters	PEINIT CSC

Table 5-26 PROFREF SU Output Data Element Requirements

Name	Description	Units	Source
h_{ref}	Heights of refractivity profile with respect to local ground height	meters	PROFREF SU
htdum	Dummy array containing height values for current (horizontally interpolated) profile	meters	PROFREF SU
lvlep	Number of height/refractivity levels in profile	N/A	PROFREF SU
n_{lvl}	Number of levels in new profile	N/A	PROFREF SU
refdum	Dummy array containing refractivity values for current (horizontally interpolated) profile	M-units	PROFREF SU
ref_{ref}	Refractivity array	M-units	PROFREF SU

5.1.13 Interpolate Profile (INTPROF) SU

The purpose of the INTPROF SU is to perform a linear interpolation vertically with height on the refractivity profile, ref_{ref}. Interpolation is performed at each PE mesh height point. Initially, the index j is set to 2.

For values of i from 0 through n_{fft}, the internal variable *height* is set equal to PE mesh height $ht(i)$. And if *height* is less than or equal to $h_{ref}(j)$, or if j is greater or equal to n_{lvl}; then the interpolated profile $profint(i)$ is determined from

$$\begin{aligned} profint(i) &= ref_{ref}(j-1) \\ &\quad + con\left(ref_{ref}\{j\} - ref_{ref}\{j-1\}\right)\frac{height - h_{ref}(j-1)}{h_{ref}(j) - h_{ref}(j-1)} \end{aligned} \tag{142}$$

where the constant con is $10^{-6}k_0$. If both of the following conditions are met: (1) *height* is less than or equal to $h_{ref}\{j\}$, and (2) j is greater or equal to n_{lvl}; then j is incremented by one and the above test is applied again. If the test is satisfied, the above interpolation is calculated, i is incremented by one, and the interpolation process begun for the new value of i. If the above test is still not satisfied, j is incremented by one until the above test is satisfied and the interpolation at i performed.

Table 5-27 and Table 5-28 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the INTPROF SU.

Table 5-27 INTPROF SU Input Data Element Requirements

Name	Description	Units	Source
con	$10^{-6} k_o$	meters^{-1}	Calling SU
h_{ref}	Heights of refractivity profile with respect to local ground height	meters	Calling SU
ht	PE mesh height array of size n_{fft}	meters	Calling SU
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	TPEM.INC
mxlvls	Maximum number of height/M-unit levels	N/A	TPEM.INC
n_{fft}	Transform size	N/A	GETFFTSZ SU
n_{lvl}	Number of levels in new profile	N/A	Calling SU
ref_{ref}	Refractivity array	M-units	Calling SU

Table 5-28 INTPROF SU Output Data Element Requirements

Name	Description	Units	Source
profint	Profile interpolated to every Δz_{PE} in height	M-units	INTPROF SU

5.2 Parabolic Equation Step (PESTEP) CSC

The purpose of the PESTEP SU is to advance the entire TPEM CSCI algorithm one output range step, referencing various SUs to calculate the propagation loss at the current output range. Figure 5-2 illustrates the general program flow for the PESTEP SU.

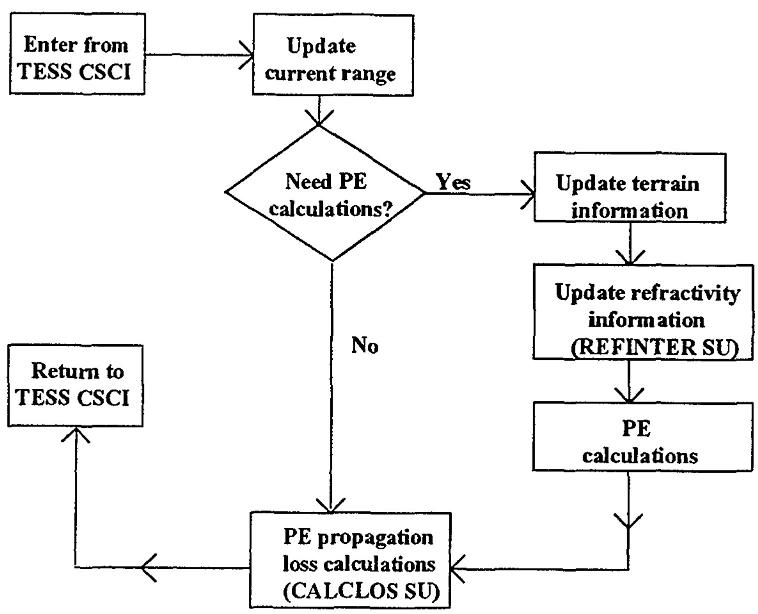

Figure 5-2 Program flow of the PESTEP CSC

Upon entering the PESTEP SU, the current PE range x_{cur} is set equal to zero if the output range x_{out} is less than or equal to 10^{-3} meters, the next output range x_{out} is determined by incrementing x_{out} by the output range step Δx_{out}. An iterative DO WHILE loop is then begun to advance the PE solution such that for the current PE range, x_{cur}, a PE solution is calculated from the solution at the previous PE solution range. This iterative procedure is repeated in the DO WHILE loop until x_{cur} is greater than the output range x_{out}.

These procedures begin as follows. First, if the saved current PE range, x_{cur}, is greater than zero, then the height of the ground at the last PE range y_{last} is set to the height of the ground at the current PE range y_{cur}. Next, the last PE range x_{last} is set equal to the current PE range x_{cur}. Then the field arrays U of the previous range step are stored in the array U_{last} for subsequent horizontal interpolation at range x_{out}. This transfer of values of U to U_{last} is made for values of the index i running from 1 through n_{fft}, the transform size. In addition, x_{cur} is incremented by the value of Δx_{PE}, the PE range step. Finally, the range at which interpolation for range-dependent refractivity profiles is performed, x_{mid}, is determined from x_{cur} and Δx_{PE2}, one-half the PE range step Δx_{PE}, as follows

$$x_{mid} = x_{cur} - \Delta x_{PE2} \ . \tag{143}$$

Then several procedures for a terrain case (i.e., *fter* is true) are instituted. First, before these procedures are begun, it is necessary to initialize the counter for the terrain profile k_t and current slope of the terrain segment, *slope*. That is, if the absolute value of the difference between the current range x_{cur} and the PE range step Δx_{PE} is less than or equal to 10^{-3}; then k_t is set to 1, and *slope* is set to the value of *slp*(1), the slope of the first segment of terrain. If the current range x_{cur} is past *tr.terx*(k_t+1), the next range point in the terrain profile, and k_t is less than the number of height/range point pairs in the profile, *tr.itp*; then k_t is incremented by one and *slope* is set to the value of *slp*(k_t). Then y_{cur}, the height of the ground at the current PE range, is given by

$$y_{cur} = tr.tery(k_t) + slope\left(x_{cur} - tr.terx\left\{k_t\right\}\right) \tag{144}$$

where *tr.tery*(k_t) is the height point of the terrain profile at index k_t. Having set the index k_p equal to the present value of k_t, the value of k_p is decreased by one in a DO WHILE

loop until x_{mid} is greater than or equal to *tr.terx*(k_p). Then the height of the ground midway between the last and current PE range, y_{curm}, is determined from

$$y_{curm} = tr.tery\left(k_p\right) + slp\left(k_p\right)\left(x_{mid} - tr.terx\left\{k_p\right\}\right) . \tag{145}$$

If vertical polarization is used (i.e., *sv.polar* equals "V"), a new complex refractive index and a new impedance term are calculated when the variable x_{cur} is greater than *tr.rgrnd*(i_g+1), the range at which the ground type i_g applies. If the latter test is satisfied, then i_g is incremented by one and a reference is made to the GETALN SU to obtain the new complex refractive index and the new impedance term. Finally, if the slope of the segment is negative, a reference to the DOSHIFT SU is made so the PE field can be "shifted" by the number of bins, or PE mesh height points, corresponding to the local ground height.

For the vertical polarization case, the difference equation *w*(*i*) of the complex PE field array is given by

$$w(i) = \alpha_v\, U(i) + \frac{U(i+1) - U(i-1)}{\Delta z_{PE2}} \tag{146}$$

where α_v is the vertical polarization impedance term, Δz_{PE2} is twice the PE mesh height, and *U* is the complex PE field. The index *i* runs from 1 through *nm1*. The parameter *nm1* is the transform size n_{fft} minus one. Using *w*(*i*), the FRSTP SU is referenced to propagate the complex PE solution field in free space by one range step. The coefficients used in vertical polarization calculations, C_1 and C_2, are propagated to the new range as follows:

$$\begin{aligned} C_1 &= C_1 * C_{1M} \\ C_2 &= C_2 * C_{2M} \end{aligned} . \tag{147}$$

If the polarization is not vertical, then the FRSTP SU is referenced using the complex PE field *U*.

If the TPEM CSCI is being used in a range-dependent mode, that is, more than one profile has been input (i.e., the number of profiles *rf.nprof* is greater than 1); or a terrain profile is specified (i.e., *fter* is true); the REFINTER SU is referenced using y_{curm}

(the height of the ground midway between the last and current PE range) to compute a new modified refractive index profile, *profint*, adjusted by the local ground height at the current range. The PHASE2 SU is then referenced to compute a new environmental phase term, *envpr*, based on this new refractivity profile.

The following steps outline the implementation of steps nine through eleven in Kuttler's formulation for vertical polarization (i.e., *sv.polar* equals "V"). First, the particular solution of Kuttler's difference equation y_m is initialized to complex zero. Then y_m is found for the index i running from 1 through *nm1* from

$$y_m\left(i\right) = \Delta z_{PE2}\, w\left(i\right) + R_T\, y_m\left(i-1\right) . \tag{148}$$

Next, the complex field $U(n_{fft})$ is set equal to complex zero. The complex field U is found with the index *nmi* decreasing from n_{fft} minus one to 0 from

$$U\left(nmi\right) = R_T\left(y_m\left\{nmi\right\} - U\left\{nmi+1\right\}\right) . \tag{149}$$

Next, the two summation terms, *sum1* and *sum2*, for determining the complex coefficients, a_r and b_r, of the partial linear solution to the homogeneous equation, respectively, are set equal to complex zero. The final values of *sum1* and *sum2* are found in a DO loop for the index i increasing from 0 through n_{fft}. The complex fields U_i and U_{nmi} are given by

$$\begin{aligned} U_i &= U\left(i\right) \\ U_{nmi} &= U\left(nmi\right) \end{aligned} \tag{150}$$

where

$$nmi = n_{nfft} - i . \tag{151}$$

If i equals either zero or n_{fft}, U_i and U_{nmi} are given by

$$\begin{aligned} U_i &= 0.5\ U_i \\ U_{nmi} &= 0.5\ U_{nmi} \end{aligned} . \tag{152}$$

The constants in the summation arguments, C_{1C} and C_{2C}, for determining a_r and b_r, respectively, are given by

$$
\begin{aligned}
C_{1C} &= U_i \; R_{AV}\left(i \right) \\
C_{2C} &= U_{nm1} \; cd
\end{aligned}
\tag{153}
$$

where *cd* is given by

$$
\begin{aligned}
cd &= R_{AV}\left(i \right) \\
cd &= -R_{AV}\left(i \right) \qquad \text{if}\left(\text{MOD}\left\{ i, 2 \right\} \equiv 1 \right)
\end{aligned}
\; .
\tag{154}
$$

Finally, *sum1* and *sum2* are given by

$$
\begin{aligned}
sum1 &= sum1 + C_{1C} \\
sum2 &= sum2 + C_{2C}
\end{aligned}
\; .
\tag{155}
$$

The constants a_r and b_r are now given by

$$
\begin{aligned}
a_r &= C_1 - R_K \; sum1 \\
b_r &= C_2 - R_K \; sum2
\end{aligned}
\tag{156}
$$

where R_K is a coefficient used in the C_1 and C_2 calculations.

In the last DO loop for Kuttler's formulation, the complex field $U(i)$ is found from the current value of $U(i)$ and $R_{AV}(i)$, $a_r(i)$ and $b_r(i)$. The loop index i runs from 0 through n_{fft}.

$$
U\left(i \right) = U\left(i \right) + a_r \; R_{AV}\left(i \right) + b_r \; cd
\tag{157}
$$

where *cd* is given by

$$
\begin{aligned}
cd &= R_{AV}\left(nmi \right) \\
cd &= -R_{AV}\left(nmi \right) \qquad \text{if}\left(\text{MOD}\left\{ nmi, 2 \right\} \equiv 1 \right)
\end{aligned}
\; .
\tag{158}
$$

and

$$
nmi = n_{nfft} - i \; .
\tag{159}
$$

The complex field $U(i)$ is now multiplied by the environmental term *envpr*(i) for values of the index i running from 1 through *nm1*.

At each PE range step, the PE field is shifted by the number of bins, or PE mesh height points, corresponding to the local ground height. This is done in a in a reference to the DOSHIFT SU.

Finally, after the output range x_{out} is reached and the DO WHILE loop exited, the CALCLOS SU is referenced to obtain the propagation loss *mloss* values at the desired output heights at the current output range x_{out}.

Table 5-29 and Table 5-30 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the GETPFAC SU.

Table 5-29 PESTEP CSC Input Data Element Requirements

Name	Description	Units	Source
α_v	Vertical polarization impedance term -> ik_o / R_{ng}	N/A	Calling CSCI
C_1	Coefficient used in vertical polarization calculations	N/A	Calling CSCI
C_2	Coefficient used in vertical polarization calculations	N/A	Calling CSCI
C_{1M}	Constant for each calculated α_v used in C_1 calculation	N/A	Calling CSCI
C_{2M}	Constant for each calculated α_v used in C_2 calculation	N/A	Calling CSCI
Δx_{out}	Output range step	meters	Calling CSCI
Δx_{PE}	PE range step	meters	Calling CSCI
Δx_{PE2}	½ PE range step	meters	Calling CSCI
Δz_{PE2}	2. * Δz_{PE}	meters	Calling CSCI
envpr	Complex refractivity profile array interpolated every Δz_{PE} in height	M-units	Calling CSCI
frsp	Complex free space propagator term array	N/A	Calling CSCI
fter	Logical flag indicating if performing terrain case	N/A	Calling CSCI
i_g	Counter indicating current ground type being modeled	N/A	Calling CSCI
maxn4	*maxpts* divided by 4; specifies the length of the filter array	N/A	TPEM.INC
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	FFTSIZ.INC
mxrout	Maximum number of output range points	N/A	TPEM.INC

Table 5-29 PESTEP CSC Input Data Element Requirements (cont'd)

Name	Description	Units	Source
mxter	Maximum number of height/range points allowed for terrain profile	N/A	TPEM.INC
mxzout	Maximum number of output height points	N/A	TPEM.INC
n_{fft}	Transform size	N/A	Calling CSCI
nm1	$n_{fft} - 1$	N/A	Calling CSCI
$R_{AV}(ii)$	Array of R_T to the *ii*th power (e.g., R_T^i)	N/A	Calling CSCI
rf	Refractivity structure for external environmental data elements	N/A	TPEM.INC
rf.nprof	Number of profiles	N/A	TPEM.INC
R_K	Coefficient used in C_1 and C_2 calculations	N/A	Calling CSCI
R_T	Complex root of quadratic equation for mixed transform method based on Kuttler's formulation	N/A	Calling CSCI
slp	Slope of each segment of terrain	meters/ meter	Calling CSCI
sv	System structure for external system data elements	N/A	TPEM.INC
sv.polar	1-character string indicating polarization	N/A	TPEM.INC
tr	Terrain structure for external system data elements	N/A	TPEM.INC
tr.itp	Number of height/range points pairs in profile	N/A	TPEM.INC
tr.rgrnd	Ranges at which the ground types apply	meters	TPEM.INC
tr.terx	Range points of terrain profile	meters	TPEM.INC
tr.trey	Height points of terrain profile	meters	TPEM.INC
U	Complex PE field	$\mu V / m$	Calling CSCI
vnp	INUTVAR structure for external implement constants	N/A	TPEM.INC
y_{cur}	Height of ground at current PE range	meters	Calling CSCI
y_{curm}	Height of ground midway between last and current PE range	meters	Calling CSCI
y_{last}	Height of ground at last PE range	meters	Calling CSCI
y_{minter}	Reference height for internal calculations of the field U (minimum height of terrain profile)	meters	Calling CSCI

Table 5-30 PESTEP CSC Output Data Element Requirements

Name	Description	Units	Source
α_v	Vertical polarization impedance term -> ik_o / R_{ng}	N/A	PESETEP CSC
C_1	Coefficient used in vertical polarization calculations	N/A	PESTEP CSC
C_2	Coefficient used in vertical polarization calculations	N/A	PESTEP CSC
C_{1M}	Constant for each calculated α_v used in C_1 calculation	N/A	PESTEP CSC
C_{2M}	Constant for each calculated α_v used in C_2 calculation	N/A	PESTEP CSC
envpr	Complex refractivity profile array interpolated every Δz_{PE} in height	M-units	PESTEP CSC
i_g	Counter indicating current ground type being modeled	N/A	PESTEP CSC
j_{end}	Index at which valid loss values in *mloss* end	N/A	PESTEP CSC
j_{start}	Index at which valid loss values in *mloss* begin	N/A	PESTEP CSC
mloss	Loss values	Centibels	PESTEP CSC
$R_{AV}(ii)$	Array of R_T to the *ii*th power (e.g., R_T^i)	N/A	PESTEP CSC
R_{ng}	Complex refractive index	M-units	PESTEP CSC
R_{ng2}	Complex refractive index squared	M-units^2	PESTEP CSC
R_T	Complex root of quadratic equation for mixed transform method based on Kuttler's formulation	N/A	PESTEP CSC
U	Complex PE field	$\mu V / m$	PESTEP CSCI
x_{out}	Output range	meters	PESTEP CSC
y_{cur}	Height of ground at current PE range	meters	PESTEP CSC
y_{curm}	Height of ground midway between last and current PE range	meters	PESTEP CSC
y_{last}	Height of ground at last PE range	meters	PESTEP CSC

5.2.1 DOSHIFT SU

The purpose of the DOSHIFT SU is to shift the field by the number of bins, or PE mesh heights corresponding to local ground height.

Upon entry, the number of bins to be shifted is determined. First, the difference y_{diff} between height of the ground at the last range step y_{last} and that at the current PE range y_{cur} is determined from

$$y_{diff} = y_{cur} - y_{last} \ . \tag{160}$$

The number of bins to be shifted k_{bin} is found from

$$k_{bin} = \mathrm{NINT}\left(\frac{\mathrm{ABS}\left\{ y_{diff} \right\}}{\Delta z_{PE}} \right) \ . \tag{161}$$

The PE solution U is then shifted downward if the local ground is currently at a positive slope ($y_{diff} > 0.0$), upward if the local ground is at a negative slope ($y_{diff} < 0.0$), and otherwise not shifted. When the PE solution has been shifted down, then the value of the PE solution U for the upper k_{bin} elements are set to zero. Likewise, when the PE solution has been shifted upwards, the lower k_{bin} elements are set to zero.

Table 5-31 and Table 5-32 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the GETPFAC SU.

Table 5-31 DOSHIFT SU Input Data Element Requirements

Name	Description	Units	Source
Δz_{PE}	Bin width in z space	meters	GETFFTSZ SU
n_{fft}	Transform size	N/A	GETFFTSZ SU
$nm1$	$n_{fft} - 1$	N/A	PEINIT CSC
U	Complex field	$\mu V / m$	Calling SU
y_{cur}	Height of ground at current range step	meters	Calling SU
y_{last}	Height of ground at last range step	meters	Calling SU

Table 5-32 DOSHIFT SU Output Data Element Requirements

Name	Description	Units	Source
U	Complex field after bin shift	$\mu V / m$	DOSHIFT SU

5.2.2 GETALN SU

The purpose of the GETALN SU is to compute the impedance term in the Leontovich boundary condition, and the complex index of refraction for finite conductivity and vertical polarization calculations. The implementation of these impedance formulas follow Kuttler's and Dockery 's method (Ref. h).

Using the user defined ground type (i_g), the relative permittivity and the conductivity input values, *tr.dielect(1,* i_g*)* and *tr.dielect(2,* i_g*)*, respectively, are used to determine the complex refractive index (R_{ng}) and complex refractive index squared (R_{ng2}) in terms of the wavelength (λ) as follows.

$$R_{ng2} = \text{CMPLX}\left(tr.dielect\left\{1, i_g\right\},\ 60.0\,\lambda\ \ tr.dielect\left\{2,\ i_g\right\}\right) \tag{162}$$

$$R_{ng} = \text{CSQRT}\left(R_{ng2}\right) \tag{163}$$

The vertical polarization impedance term (α_v) is given in terms of the complex refractive index (R_{ng}), the imaginary (i), and free space wave number (k_o) by

$$\alpha_v = \frac{ik_o}{R_{ng}}\ . \tag{164}$$

The determination of the complex root (R_T) of the quadratic equation for the mixed transform method is based on Kuttler's formulation. It is done here only for vertical polarization, as that is the only condition in which the GETALN SU will be called. First, the internal parameter, R_T is determined as follows.

$$R_T = \text{CSQRT}\left(1.0 + \left\{\alpha_v\ \Delta z_{PE}\right\}^2\right) - \alpha_v\ \Delta z_{PE} \tag{165}$$

Then, for values of 0 through n_{ffi} of the index ii, the array of (R_T) to the ii^{th} power $R_{AV}(ii)$ is given by

$$R_{AV}(ii) = R_T^{ii}\ . \tag{166}$$

The parameter R_K, a coefficient used in the determination of C_1 and C_2 in the calling SU, is found from

$$R_K = \frac{2.0\left(1.0 - R_{AV}\{2\}\right)}{\left(1.0 - R_{AV}^2\left\{n_{flt}\right\}\right)\left(1.0 + R_{AV}\{2\}\right)} \quad . \tag{167}$$

The parameters C_{1M} and C_{2M} are determined from

$$C_{1M} = \text{CEXP}\left(\frac{i\Delta x_{PE}}{2k_0}\left\{\frac{\text{CLOG}(R_T)}{\Delta z_{PE}}\right\}^2\right) \quad . \tag{168}$$

and

$$C_{2M} = \text{CEXP}\left(\frac{i\Delta x_{PE}}{2k_0}\left\{\frac{\text{CLOG}\left(R_T\right) - \pi i}{\Delta z_{PE}}\right\}^2\right) \quad . \tag{169}$$

Table 5-33 and Table 5-34 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the GETALN SU.

Table 5-33 GETALN SU Input Data Element Requirements

Name	Description	Units	Source
Δx_{PE}	PE range step	meters	Calling SU
Δz_{PE}	Bin width in z space	meters	GETFFTSZ SU
i	Imaginary i = complex (0,1)	N/A	PEINIT CSC
i_g	Counter indicating current ground type being modeled	N/A	CALLING SU
k_o	Free-space wave number $= 2\pi / \lambda$	meters^{-1}	PEINIT CSC
λ	wave length	meters	PEINIT CSC
$maxpts$	Maximum size of arrays for the real and imaginary fields	N/A	FFTSIZ.INC
n_{fft}	Transform size	N/A	GETFFTSZ SU
π	3.1415926	N/A	TPEM.INC
tr	Terrain structure for external terrain data elements	N/A	TPEM.INC
$tr.dielec$	2-dimensional array containing the relative permittivity and conductivity for user defined terrain	$^{\alpha}$	TPEM.INC

$^{\alpha}$ conductivity has units of S/m.

Table 5-34 GETALN SU Output Data Requirements

Name	Description	Units	Source
α_v	Vertical polarization impedance term -> ik_o / R_{ng}	N/A	GETALN SU
C_{1M}	Constant for each calculated α_v used in C_1 calculation	N/A	GETALN SU
C_{2M}	Constant for each calculated α_v used in C_2 calculation	N/A	GETALN SU
$R_{AV}(ii)$	Array of R_T to the iith power (e.g., R_T^i)	N/A	GETALN SU
R_K	Coefficient used in C_1 and C_2 calculations	N/A	GETALN SU
R_{ng}	Complex refractive index	N/A	GETALN SU
R_{ng2}	Complex refractive index squared	N/A	GETALN SU
R_T	Complex root of quadratic equation for mixed transform method based on Kuttler's formulation	N/A	GETALN SU

5.2.3 Free Space Range Step (FRSTP) SU

The purpose of the FRSTP SU is to propagate the complex PE solution field in free space by one range step.

Upon entry the PE field, *farray*, is transformed to p-space (Fourier space) and then *nm1* of its elements are multiplied by the corresponding free space propagator, *frsp*, elements. Before exiting the PE field is transformed back to z-space. Both transforms are performed using FFT SU.

Table 5-35 and Table 5-36 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the FRSTP SU.

Table 5-35 FRSTP SU Input Data Element Requirements

Name	Description	Units	Source
farray	Field array to be propagated one range step in free space	$\mu V / m$	Calling SU
frsp	Complex free space propagator term array	N/A	Calling SU
nm1	*n-1*	N/A	PEINIT CSC

Table 5-36 FRSTP SU Output Data Element Requirements

Name	Description	Units	Source
farray	Propagated field array	$\mu V / m$	FRSTP SU

5.2.4 Refractivity Interpolation (REFINTER) SU

The purpose of the REFINTER SU is to interpolate both horizontally and vertically on the modified refractivity profiles. Profiles are then adjusted so they are relative to the local ground height .

An in-line function for linear interpolation is defined by

$$\mathrm{PINT}\,(p1, p2) \;=\; p1 \;+\; fv\;(p2 \;-\; p1)\;\;. \tag{170}$$

Upon entry, the number of height/refractivity levels, *lvlep*, in. the profile is set equal to the user input number of levels, *rf.lvlep*, in the refractivity profile. For the range dependent case all profiles have the same number of levels.

If there is a range-dependent environment (i.e., more than one refractivity profile), horizontal interpolation in range to the current PE range is performed between the two neighboring profiles that are specified relative to mean sea level. If the user input parameter, *rf.nprof*, is greater than one, then the case is range dependent. In that case the following calculations are made. First, if the desired range for profile interpolation, *range*, is greater than the range for the next refractivity profile *rv2*; then the index *j* of the last refractivity profile is set equal to the counter for the current profile i_s, i_s is incremented by one, the range of the last refractivity profile *rv1* is set equal to *rv2*, *rv2* is set equal to the range of the $i_s^{\,th}$ profile *rf.rngprof*(i_s). The fractional range *fv* for the interpolation is given by

$$fv \;=\; \frac{range \;-\; rv1}{rv2 \;-\; rv1}\;\;. \tag{171}$$

For values of the index *i* from 1 to *lvlep*, the dummy array *refdum* containing M-unit values for the current (interpolated) profile and the dummy array *htdum* containing height values for the current (interpolated) profile are determined from

$$refdum(i) \;=\; \mathrm{PINT}\left(rf.refmsl\{i,j\},\, rf.refmsl\{i,i_s\}\right) \tag{172}$$

and

$$htdum(i) \; = \; \text{PINT}\left(rf.hmsl\{i,j\}, rf.hmsl\{i,i_s\} \right) \tag{173}$$

where *rf.refmsl* is a 2-dimensional array containing refractivity with respect to mean sea level of each profile and *rf.hmsl* is a 2-dimensional array containing heights in meters with respect to mean sea level of each profile.

The REMDUP SU is referenced to remove duplicate refractivity levels with *lvlep* being the number of points in the profile at range *range*, and the PROFREF SU is then referenced to adjust the new profile (i.e., *refdum* and *htdum*) relative to the internal reference height h_{minter} corresponding to the minimum height of the terrain profile. The PROFREF SU is then referenced once more to adjust the profile relative to the local ground height y_{curm}, and upon exit from the PROFREF SU, the INTPROF SU is referenced to interpolate vertically on the refractivity profile at each PE mesh height point. This results in the n_{fft}-point profile *profint*() array containing the interpolated M-unit values for the refractivity at range *range*, where n_{fft} is the transform size.

Upon exiting the REFINTER SU, *rv1* and the index *j* are saved. The index *j* is the index of the last refractivity profile.

Table 5-37 and Table 5-38 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the REFINTER SU.

Table 5-37 REFINTER SU Input Data Elements

Name	Description	Units	Source
i_s	Counter for current profile	N/A	Calling SU
lvlep	Number of height/refractivity levels in profile	N/A	Calling SU
mxlvls	Maximum number of height/M-unit levels	N/A	TPEM.INC
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	FFTSIZ.INC
range	Range for profile interpolation	meters	Calling SU
rf	Refractivity structure for external environmental data elements	N/A	TPEM.INC
rf.lvlep	Number of levels in refractivity profile	N/A	TPEM.INC
rf.refmsl	2-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be *refmsl(i,j)* = M-unit at i^{th} level of j^{th} profile. j=1 for range-independent cases	M-unit	TPEM.INC
rf.hmsl	2-dimensional array containing heights with respect to mean sea level of each profile. Array format must be *hmsl(i,j)* = height of i^{th} level of j^{th} profile. j=1 for range-independent cases	meters	TPEM.INC
rf.rngprof	Ranges of each profile. *rngprof(i)* = range of i^{th} profile	meters	TPEM.INC
rf.nprof	Number of profiles	N/A	TPEM.INC
rv2	Range of the next refractivity profile	meters	Calling SU
y_{curm}	Height of ground midway between last and current range step	meters	Calling SU
y_{minter}	Reference height for internal calculations of the field U	meters	Calling SU

Table 5-38 REFINTER SU Output Data Element Requirements

Name	Description	Units	Source
i_s	Counter for current profile	N/A	REFINTER SU
lvlep	Number of height/refractivity levels in profile	N/A	REFINTER SU
profint	Profile interpolated to every Δz_{PE} in height	M-units	REFINTER SU
rv2	Range of the next refractivity profile	meters	REFINTER SU

5.2.5 Remove Duplicate Refractivity Levels (REMDUP) SU

The purpose of the REMDUP SU is to remove any duplicate refractivity levels in the currently interpolated profile. Adjoining profile levels are checked to see if the heights are within 0.001 meters. If they are, the duplicate level in the profile is removed. This process continues until all profile levels (*lvlep*) have been checked.

Table 5-39 and Table 5-40 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the REMDUP SU.

Table 5-39 REMDUP SU Input Data Element Requirements

Name	Description	Units	Source
htdum	Dummy array containing height values for current (horizontally interpolated) profile	meters	REFINTER SU
lvlep	Number of height/refractivity levels in profile	N/A	REFINTER SU
mxlvls	Maximum number of height/M-unit levels	N/A	TPEM.INC SU
refdum	Dummy array containing M-unit values for current (horizontally interpolated) profile	M-units	REFINTER SU

Table 5-40 REMDUP SU Output Data Element Requirements

Name	Description	Units	Source
htdum	Dummy array containing height values for current (horizontally interpolated) profile	meters	REMDUP SU
lvlep	Number of height/refractivity levels in profile	N/A	REMDUP SU
refdum	Dummy array containing M-unit values for current (horizontally interpolated) profile	M-units	REMDUP SU

5.2.6 Calculate Propagation Loss (CALCLOS) SU

The purpose of the CALCLOS SU is to determine the propagation loss at each output height point at the current output range.

At the outset a minimum propagation factor, *pfacmin*, is set to 300 dB.

Then an in-line function for linear interpolation between two values pl_1 and pl_2, PLINT, is defined by

$$\mathrm{PLINT}\left(pl_1, pl_2, frac\right) = pl_1 + frac\left(pl_2 - pl_1\right) \tag{174}$$

where *frac* is the fractional distance from pl_1 to pl_2 for which the interpolation is being made.

Several variables are initialized. If the output range step Δx_{out} is within 10^{-3} meters of the output range x_{out}; the counter, i_c, for the array that contains heights y_{lim} at each output range at which the last valid loss value exists, is set to 1. The height of the terrain, y_{ch}, at the current range step is determined relative to the reference height, y_{mref}. The reference height, y_{mref}, is itself a user provided minimum output height relative to the minimum terrain height, y_{minter} (e.g., sea level). Then, the height of the terrain, y_{ct}, at the current range step relative to y_{minter} is determined. Next, the height of the terrain, y_{lh}, at the last range step is determined relative to y_{mref}. Finally, the height of the terrain, y_{lt}, at the last range step relative to the minimum terrain height, y_{minter}, is determined.

The interpolated ground height z_{int} at the current output range x_{out} and the number of vertical output points i_{zg} that correspond to this ground height are determined. First, the interpolated ground height is given by

$$z_{int} = \mathrm{PLINT}\left(y_{last}, y_{cur}, xx\right) \tag{175}$$

where the parameter xx is given in terms of the PE range step Δx_{PE} by

$$xx = \frac{x_{out} - x_{last}}{\Delta x_{PE}} . \tag{176}$$

Having determined z_{int}, then i_{zg} is determined from

$$i_{zg} = \mathrm{INT}\left(\frac{z_{\mathrm{int}} - y_{mref}}{\Delta z_{out}} \right) . \tag{177}$$

where Δz_{out} is the output height increment. Then all of the elements of the array *mloss* from 1 to i_{zg} are set to zero, and the index, j_{start}, representing beginning valid loss values in the *mloss* array is set to the maximum of 1 or $\left(i_{zg}+1\right)$.

If the current output range x_{out} is greater than the range at which valid loss values are to be calculated, then the calculation of loss values and the *mloss* array is begun. If this condition is not satisfied, then the *mloss* array is set to -1 for values of the array index from a beginning value of j_{start} up to and including the number of output points desired *vnp.nzout*.

First, several parameters needed in the determination of the propagation loss are calculated. If the last PE range, x_{last}, is greater than zero, then the parameter, x_{loglst}, is given by

$$x_{loglst} = 10.\ \mathrm{ALOG10}\left(x_{last}\right) \qquad for\ x_{last} > 0. \tag{178}$$

Otherwise x_{loglst} is set to zero. The parameter x_{log} is given by

$$x_{log} = 10.\ \mathrm{ALOG10}\left(x_{PE}\right) \tag{179}$$

where x_{PE} is the PE range. The free space loss, $fspace_{x_{out}}$, at x_{out} is given by

$$fspace_{x_{out}} = 20.\ \mathrm{ALOG10}\left(x_{out}\right) + plcnst\ . \tag{180}$$

Several parameters are determined corresponding to the terrain case. If the logical variable *fter* is true, then a terrain case is being calculated. The two indices i_{p1} and i_{p2} are given by

$$i_{p1} = \mathrm{AMAX0}\left(0, \mathrm{INT}\left\{ \frac{y_{lh}}{\Delta z_{out}} \right\}\right) \tag{181}$$

and

$$i_{p2} = \text{AMAX0}\left(0, \text{INT}\left\{\frac{y_{ch}}{\Delta z_{out}}\right\}\right) . \tag{182}$$

For values of the array index from 1 up to and including i_{p1}, the array of propagation factors, *rfac1*, at valid height points for range x_{last} are set to the minimum propagation factor *pfacmin* for later interpolation. For values of the array index from 1 up to and including i_{p2}, the array of propagation factors, *rfac2*, at valid height points for range x_{out} are set to the minimum propagation factor *pfacmin* for later interpolation. Then the two indices i_{p1} and i_{p2} are incremented by a value of 1.

If the logical variable *fter* is false (i.e., a smooth surface case), then both i_{p1} and i_{p2} are set to 1.

Next, the height/integer value, j_{end}, to stop calculating the loss is determined. j_{end} is found as follows.

$$j_{end} = \text{AMAX0}\left(0, \text{NINT}\left\{\frac{\text{AMIN1}\left[z_{lim}, \text{AMAX1}\left(z_{int}, y_{lim}\{i_c\}\right)\right] - y_{mref}}{\Delta z_{out}}\right\}\right) \tag{183}$$

where i_c is the counter for the array $y_{lim}(i)$. Note that for terrain cases, ray tracing was performed using the direct ray angle and sometimes $y_{lim}(i)$ may be less than the local ground height. In that case this SU exits from the propagation loss calculation using a GO TO FORTRAN statement.

The propagation loss values are determined from the propagation factors *rfac1*(i) and *rfac2*(i) and from the parameter *xx* shown above. If x_{loglst} is greater than zero, then the propagation factor, *rfac1*(i), at valid heights from the field at the previous step U_{last} is given by

$$rfac1(i) = \text{GETPFAC}\left(U_{last}, x_{\log}, z_{out}\{i\} - y_{lt}\right) \tag{184}$$

where the index i goes from i_{p1} to j_{end} in steps of 1. Then, using a reference to the GETPFAC SU, the propagation factor, *rfac2(i)*, at valid heights from the field U is given by

$$rfac2(i) = \text{GETPFAC}\left(U, x_{log}, z_{out}\{i\} - y_{ct} \right) \tag{185}$$

where the index i goes from i_{p2} to j_{end} in steps of 1. Finally, the propagation loss *mloss* at range x_{out} is found by interpolating between the two PE ranges. For the index k running from j_{start} to j_{end} in steps of one and using a reference to the in-line function PLINT, the propagation loss *mloss*(k) is given by

$$mloss(k) = \text{INT2}\left(10. \left\{ \text{PLINT}\left[rfac1(k), rfac2(k), xx \right] + fspace_{x_{out}} \right\} \right) \tag{186}$$

when x_{loglst} is greater than zero. Otherwise, *mloss*(k) is given by

$$mloss(k) = \text{INT2}\left(10. \left\{ rfac2(k) + fspace_{x_{out}} \right\}\right) . \tag{187}$$

Before exiting the CALCLOS SU, the index i_c is incremented by one.

Table 5-41 and Table 5-42 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the CALCLOS SU.

Table 5-41 CALCLOS SU Input Data Element Requirements

Name	Description	Units	Source
Δx_{out}	Output range step	meters	Calling SU
Δx_{PE}	PE range step	meters	Calling SU
Δz_{out}	Output height increment	meters	Calling SU
fter	Logical flag representing terrain	N/A	PEINIT CSC
maxpts	Maximum size of arrays for the real and imaginary fields	N/A	FFTSIZ.INC
mxrout	Maximum number of output range points	N/A	TPEM.INC
mxzout	Maximum number of output height points	N/A	TPEM.INC
plcnst	Constant used in determining propagation loss $\left(20\log\left\{2k_o\right\}\right)$	N/A	PEINIT CSC
x_{PE}	PE range	meters	Calling SU
x_{last}	Last PE range	meters	Calling SU
x_{out}	Output range	meters	Calling SU
U	Complex field at current PE range	$\mu V / m$	Calling SU
U_{last}	Complex field at previous PE range step	$\mu V / m$	Calling SU
vnp	*INPUTVAR* structure for external implementation constants	N/A	Calling SU
vnp.nzout	Integer number of output height points desired	N/A	TPEM.INC
y_{cur}	Height of ground at current PE range step	meters	Calling SU
y_{last}	Height of ground at last range PE step	meters	Calling SU
y_{lim}	Height at each output range at which the last valid loss value exists	meters	Calling SU
y_{minter}	Reference height for internal calculations of the field U	meters	Calling SU
y_{mref}	Height relative to y_{minter}	meters	Calling SU
z_{lim}	Height limit for ray trace	meters	Calling SU
z_{out}	Array containing all output height points	meters	Calling SU

Table 5-42 CALCLOS SU Output Data Element Requirements

Name	Description	Units	Source
j_{end}	Index at which valid loss values in *mloss* end	N/A	CALCLOS SU
j_{start}	Index at which valid loss values in *mloss* begin	N?A	CALCLOS SU
mloss	Loss values	Centibels	CALCLOS SU

5.2.7 Get Propagation Factor (GETPFAC) SU

The purpose of the GETPFAC SU is to determine the propagation factor at the specified height in dB.

A linear interpolation over the PE range interval, Δz_{PE}, is performed to obtain the propagation factor f_{fac} for each PE height output point at the current range. First, the interpolated amplitude *pow* of the field at the receiver height, z_r, is determined from

$$pow = \mathrm{CABS}\left(U\{n_b\}\right) + f_r\left(\mathrm{CABS}\left\{U[n_b+1]\right\} - \mathrm{CABS}\left\{U[n_b]\right\}\right) \tag{188}$$

where the interpolation fraction f_r is determined from

$$f_r = \frac{z_r}{\Delta z_{PE}} - \mathrm{FLOAT}\left(n_b\right) \tag{189}$$

and where n_b is determined from

$$n_b = \mathrm{INT}\left(\frac{z_r}{\Delta z_{PE}}\right) . \tag{190}$$

pow is constrained to be greater or equal to 10^{-13} $\mu V / m$. Finally, the propagation factor f_{fac} is given by

$$f_{fac} = -20\,\mathrm{ALOG10}\left(pow\right) - x_{log} \tag{191}$$

where x_{log} is ten times the logarithm of the range.

Table 5-43 and Table 5-44 identify, describe the purpose for, state the units of, and show the computational source for each input and output data element, respectively, of the GETPFAC SU.

Table 5-43 GETPFAC SU Input Data Element Requirements

Name	Description	Units	Source
Δz_{PE}	Bin width in z space	meters	Calling SU
x_{log}	10 times logarithm of range	dB/meter	CALCLOS SU
U	Complex field	$\mu V / m$	CALCLOS SU
z_r	Receiver height	meters	CALCLOS SU

Table 5-44 GETPFAC SU Output Data Element Requirements

Name	Description	Units	Source
f_{fac}	Propagation factor at specified height	dB	GETPFAC SU

6. REQUIREMENTS TRACEABILITY

This section provides the traceability of the design of the TPEM CSCI. Table 6-1 presents this traceability between the corresponding sections of the Software Requirements Specification (SRS) and the Software Design Description (SDD) and between the various components of the TPEM CSCI.

Table 6-1 Traceability Matrix between the SRS and the SDD

Software Requirements Specification		**Software Design Description**	
SRS Requirement Name	SRS Paragraph Number	Software Design Description Name	SDD Paragraph Number
CSCI Capability Requirements	3.1	CSCI-WIDE DESIGN DECISIONS	3.
CSCI Capability Requirements	3.1	CSCI Components	4.1
CSCI Capability Requirements	3.1	Concept of Execution	4.2
Parabolic Equation Initialization (PEINIT) CSC	3.1.1	Parabolic Equation Initialization (PEINIT) CSC	5.1
Antenna Pattern (ANTPAT) SU	3.1.1.1	Antenna Pattern (ANTPAT) SU	5.1.1
Refractivity Initialization (REFINIT) SU	3.1.1.2	Refractivity Initialization (REFINIT) SU	5.1.2
Trace for Minimum Angle (TRACEA SU)	3.1.1.3	Trace for Minimum Angle (TRACEA SU)	5.1.3
Dielectric Initialization (DIEINIT) SU	3.1.1.4	Dielectric Initialization (DIEINIT) SU	5.1.4
Get FTT Size (GETFFTSZ) SU	3.1.1.5	Get FTT Size (GETFFTSZ) SU	5.1.5
Starter Field Initialization (XYINIT) SU	3.1.1.6	Starter Field Initialization (XYINIT) SU	5.1.6
Fast-Fourier Transform (FFT) SU	3.1.1.7	Fast-Fourier Transform (FFT) SU	5.1.7
Sine Fast-Fourier Transform (SINFFT) SU	3.1.1.8	Sine Fast-Fourier Transform (SINFFT) SU	5.1.8
Trace Launch Angle (TRACEH) SU	3.1.1.9	Trace Launch Angle (TRACEH) SU	5.1.9

Table 6-1 Traceability Matrix between the SRS and the SDD (con't)

Software Requirements Specification		**Software Design Description**	
SRS Requirement Name	SRS Paragraph Number	Software Design Description Name	SDD Paragraph Number
Free-Space Propagator Phase Term (PHASE1) SU	3.1.1.10	Free-Space Propagator Phase Term (PHASE1) SU	5.1.10
Environmental Propagator Phase Term (PHASE2) SU	3.1.1.11	Environmental Propagator Phase Term (PHASE2) SU	5.1.11
Profile Reference (PROFREF) SU	3.1.1.12	Profile Reference (PROFREF) SU	5.1.12
Interpolate Profile (INTPROF) SU	3.1.1.13	Interpolate Profile (INTPROF) SU	5.1.13
Parabolic Equation Step (PESTEP) SU	3.1.2	Parabolic Equation Step (PESTEP) SU	5.2
DOSHIFT SU	3.1.2.1	DOSHIFT SU	5.2.1
GETALN SU	3.1.2.2	GETALN SU	52.2
Free Space Range Step (FRSTP) SU	3.1.2.3	Free Space Range Step (FRSTP) SU	5.2.3
Refractivity Interpolation (REFINTER) SU	3.1.2.4	Refractivity Interpolation (REFINTER) SU	5.2.4
Remove Duplicate Refractivity Levels (REMDUP) SU	3.1.2.5	Remove Duplicate Refractivity Levels (REMDUP) SU	5.2.5
Calculate Propagation Loss (CALCLOS) SU	3.1.2.6	Calculate Propagation Loss (CALCLOS) SU	5.2.6
Get Propagation Factor (GETFAC) SU	3.1.2.7	Get Propagation Factor (GETFAC) SU	5.2.7
CSCI External Interface Requirements	3.2	External Interface	4.3.2
CSCI Internal Interface Requirements	3.3	Internal Interface	4.3.3
CSCI Internal Data Requirements	3.4	Internal Data	4.3.4

Table 6-1 Traceability Matrix between the SRS and the SDD (con't)

Software Requirements Specification		**Software Design Description**	
SRS Requirement Name	SRS Paragraph Number	Software Design Description Name	SDD Paragraph Number
Environmental Radio Refractivity field Data Element	3.5.1	Environmental Radio Refractivity field Data Element	7.2
Terrain Profile Data Element	3.5.2	Terrain Profile Data Element	7.3
Implementation and Application Considerations	3.10.1	Implementation and Application Considerations	7.1

7. NOTES

7.1 TPEM CSCI Implementation And Application Considerations

The calling TESS CSCI application will determine the employment of the TPEM CSCI. However, the intensive computational nature of the TPEM CSCI must be taken into consideration when designing an efficient calling application. For this reason, the TPEM CSCI is designed with flexibility for various hardware suites and computer resource management considerations. This TPEM CSCI applies only to a coverage and loss diagram application. The following highly recommended guidelines are provided to aid in the design of a coverage or loss diagram application which will most efficiently employ the TPEM CSCI.

The TPEM CSCI propagation loss calculations are independent of any target or receiver considerations, therefore, for any EM emitter, one execution of the TPEM CSCI may be used to create both a coverage diagram and a loss diagram. Since both execution time and computer memory allocation are a consideration when employing this model, it is most efficient and appropriate to execute the TPEM CSCI for a particular EM system/environmental/terrain combination before executing any application. The output of the TPEM CSCI is stored in a file which can be accessed by multiple applications.

For example, the TESS operator may desire a coverage diagram for one particular radar system. At the beginning of the coverage diagram application, a check would be made for the existence of a previously created TPEM CSCI output file appropriate for the EM system, environmental, and terrain conditions. If such a file exists, the propagation loss values can be read from the file and used to create the coverage diagram. If the file does not exist, the TPEM CSCI would be executed to create one. As the TPEM CSCI is executing, its output could be routed simultaneously to a graphics display device and a file. This file could then be used in the loss diagram application should the operator also choose it. Two distinct applications therefore, are achieved with only one execution of the TPEM CSCI. Additionally, should the operator desire an individual coverage diagram for each of multiple targets, or a single coverage diagram illustrating radar detection of a low-flying missile superimposed upon a coverage diagram illustrating his own radar's vulnerability as defined by the missile's ESM receiver, only a single execution of the TPEM CSCI would be required, thereby saving valuable computer resources.

7.2 Environmental Radio Refractivity Field Data Elements

The radio-refractivity field, i.e. the profiles of M-units versus height, should consist of vertical piece-wise linear profiles specified by couplets of height in meters above mean sea level and modified refractivity (M-units) at multiple arbitrary ranges. All vertical profiles must contain the same number of vertical data points, and be specified such that each numbered data point corresponds to like-numbered points (i.e. features) in the other profiles. The first numbered data point of each profile must correspond to a height of zero mean sea level and the last numbered data point must correspond to a height such that the modified refractivity for all greater heights is well represented by extrapolation using the two highest profile points specified. Within each profile, each numbered data point must correspond to a height greater than or equal to the height of the previous data point. Note that this requirement allows for a profile which contains redundant data points. Note also that all significant features of the refractivity profiles must be specified, even if they are above the maximum output height specified for a particular application of TPEM.

The TESS CSCI application designer and the TESS operator share responsibility for determining appropriate environmental inputs. For example, a loss diagram may be used to consider a surface-to-surface radar detection problem. Since the operator is interested in surface-to-surface, he may truncate the profile assuming that effects from elevated ducting conditions are negligible. It may be however, that the elevated duct does indeed produce a significant effect. The operator should insure therefore, that the maximum height of the profile allows for the inclusion of all significant refractive features.

This specification allows a complicated refractivity field to be described with a minimum of data points. For example, a field in which a single trapping layer linearly descends with increasing range can be described with just two profiles containing only four data points each, frame (a) of Figure 7-1. In the same manner, other evolutions of refractive layers may be described. Frames (b) and (c) of Figure 7-1 show two possible scenarios for the development of a trapping layer. The scenario of choice is the one which is consistent with the true thermodynamical and hydrological layering of the atmosphere.

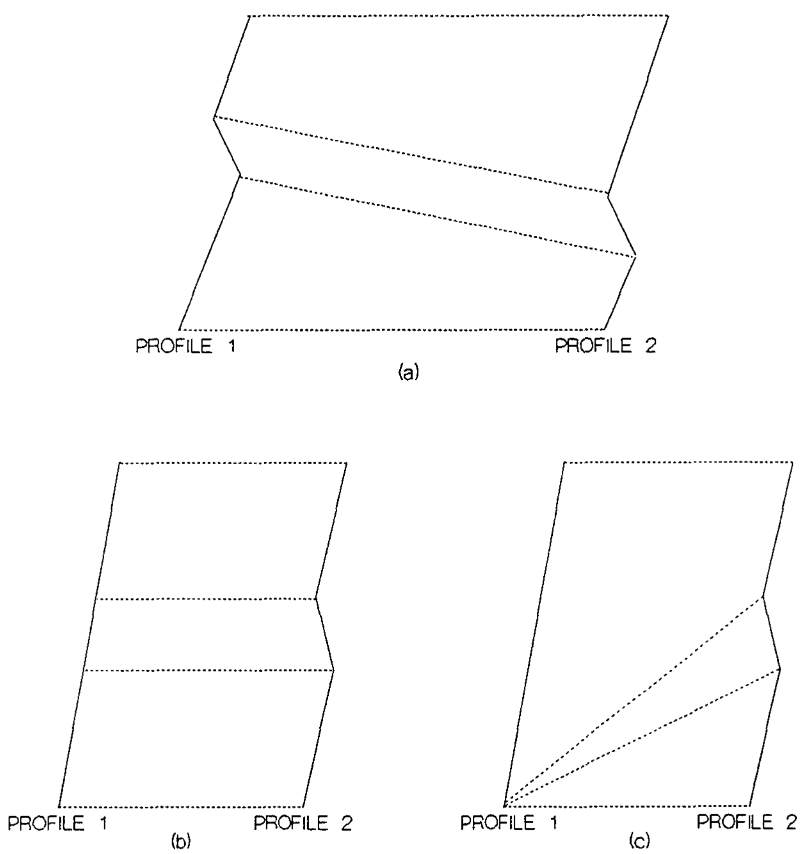

Figure 7-1 Idealized M-unit profiles (solid) and lines of interpolation (dashed)

The two TPEM CSCI implementation constants *mxlvls* and *mxnprof* refer to the maximum number of height levels allowed within a profile and the maximum number of profiles allowed by the TPEM CSCI. These two constants must be specified when the TPEM CSCI is compiled and be carefully chosen to be just large enough for all calling applications but small enough to efficiently conserve both computer memory and execution time of the TPEM CSCI. While there is no upper limit on *mxlvls* imposed by the TPEM CSCI, increasing the number of environmental levels will increase the TPEM CSCI execution time. Increasing the execution time for any particular application denies valuable computer resources to other applications, and thus makes the application less likely to be used by an operator.

Two external implementation data variables applicable to both the TESS operator and to the calling application designer are x_{max}, the maximum TPEM CSCI output range, and y_{max}, the maximum TPEM CSCI output height. These two parameters are required by the TPEM CSCI to determine the horizontal and vertical resolution respectively for internal range and height calculations based on the current values of *vnp.nrout* and *vnp.nzout*. Any value of x_{max} and y_{max} is allowed for the convenience of the TESS operator and the calling application designer. For example, the TESS operator may desire a coverage diagram which extends to a range of 500 kilometers (km). In addition to accommodating the desires of the operator, specification of such a convenient maximum range eases the burden for the application designer in determining incremental tick marks for the horizontal axis of the display.

Provided the value of the parameter *ef.lerr12* is set to '.false.', if the furthest environment profile range is less than x_{max}; the TPEM CSCI will automatically create an environment profile at x_{max} equal to the last profile specified, making the environment homogeneous from the range of the last profile specified to x_{max}. For example, a profile is input with an accompanying range of 450 km. If the TESS operator chooses an x_{max} of 500 km, the TPEM CSCI will continue loss calculations to 500 km, keeping the refractivity environment homogeneous from 450 km to 500 km.

If *ef.lerr12* is set to '.true.' and the furthest environment profile range is less than x_{max}, then an error will be returned in *ierror* from the PEINIT CSC. This is to allow the TESS CSCI application designer greater flexibility in how environment data is handled.

7.3 Terrain Profile Data Element

The terrain profile should consist of linear piece-wise segments specified in terms of range/height pairs. The TPEM CSCI implementation constant *mxter* refers to the maximum number of height/range pairs allowed within a terrain profile. This constant must be specified when the TPEM CSCI is compiled and must be carefully chosen to be just large enough for all calling applications but small enough to efficiently conserve both computer memory and execution time of the TPEM CSCI. While there is no upper limit on *mxter* imposed by the TPEM CSCI, increasing the number of terrain points will increase the TPEM CSCI execution time. Increasing the execution time for any particular

application denies valuable computer resources to other applications, and thus makes the application less likely to be used by an operator.

All range values must be increasing, and the first terrain height value must be at range zero. General ground composition types can be specified, along with corresponding ranges over which the ground type is to be applied. If ground type "User Defined" is specified (*tr.igrnd*() = 5), then numeric values of relative permittivity and conductivity must be given. If horizontal antenna polarization is specified, the TPEM CSCI will assume perfect conductivity for the entire terrain profile and will ignore any information regarding ground composition. If vertical antenna polarization is specified, then information regarding ground composition must also be specified.

The maximum height, y_{max}, must always be greater than the minimum height, y_{min}. Also, a value of y_{max}, must be given such that it is larger than the maximum elevation height along a specified terrain profile.

Provided *ef.lerr6* is set to '.false.', if the furthest range point in the terrain profile is less than x_{max}, the TPEM CSCI should automatically create a height/range pair as part of the terrain profile at x_{max} with elevation height equal to the last height specified in the profile, making the terrain profile flat from the range of the last profile point specified to x_{max}. For example, a terrain profile is input where the last height/range pair is 50 meters (m) in height with an accompanying range of 95 km. If the TESS operator chooses an x_{max} of 100 km, the TPEM CSCI should continue loss calculations to 100 km, keeping the terrain profile flat from 95 km to 100 km with an elevation height of 50 m.

If *ef.lerr6* is set to '.true.' and the furthest range point is less than x_{max}, then an error should be returned in *ierror* from the PEINIT CSC. This is to allow the TESS CSCI application designer greater flexibility in how terrain data is handled.

7.4 Acronym and Abbreviations

The following table, Table 7-1, is a glossary of acronyms and abbreviations used within this document.

Table 7-1 Acronyms and Abbreviations

Term	Definition
ABS	Absolute value
ALOG10	Logarithm to base 10
AMIN0	Minimum of integer variables
AMIN1	Minimum of real variables
AMAX0	Maximum of integer variables
AMAX1	Maximum of real variables
Centibel	One-hundredth of the logarithm of a quantity
CEXP	Exponent of complex number
CLOG	Natural logarithm of complex number
CMPLX	Data conversion to complex number
CABS	Absolute value of complex number
COMMON BLOCK	Allows two or more FORTRAN Sus to share variables without having to pass them as arguments
CONJG	Conjugate of complex number
CSCI	Computer software configuration item
CSQRT	Square root of complex number
DATA	Assigns initial values to variables
dB	Decibel
decibel	10 times the logarithm of a quantity
EM	electromagnetic
FFT	Fast-Fourier Transform
FLOAT	Data conversion from integer to floating point
FORTRAN	Formula Translation
IMAG	Imaginary part of complex number
INT	Integer value of
INT2	2 byte integer value of
km	Kilometers
LOG	Natural logarithm
m	Meters
M	Modified refractivity units

Table 7-1 Acronyms and abbreviations (con't)

Term	Definition
MHz	MegaHertz
M-unit	Refractivity meaurement unit
$\mu V / m$	Microvolts per meter
N/A	Not applicable
NINT	Round real number
p space	Phase space
radian	Unit of angular measurement
PE	Parabolic Equation
REAL	Real part of complex number
SIN	Sine function
S/m	Conductivity unit Siemans per meter
Sin(X)/X	Sine(X)/X
SRS	Software Requirements Specification
SU	Software unit
TPEM	Terrain Parabolic Equation Model
TESS	Tactical Environmental Support System

7.5 SDD Variable Name, FORTRAN Variable Name Cross Reference

The following table, Table 7-2, is a cross reference of variable names used within the body of this document and the FORTRAN variable names as used within the TPEM CSCI source code of Section 8, Appendix A. Included are the SDD variable name, its description, the FORTRAN source code name, and the designation of the FORTRAN COMMON BLOCK name if applicable.

Table 7-2 Variable name cross reference

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
a	Argument of internal function in TRACEA SU	a	N/A
a	Internal variable in TRACEA SU	a	N/A
a_0	Angle of the ray before trace step	a0	N/A
a_1	Angle of the ray after trace step	a1	N/A
a_{crit}	Critical angle, angle above which no rays are trapped for ducting environment	acrit	N/A
	Specified elevation angle	u	N/A
α_1	Critical angle determined from ant_{ref} and above	acrit1	N/A
α_2	Critical angle determined from ant_{ref} and below	acrit2	N/A
α_v	Vertical polarization impedance term -> ik_o / R_{ng}	alphav	impedance
α_{pat}	Adjusted antenna elevation angle	udif	N/A
a_{mxcur}	Maximum of local angle along ray	amxcur	N/A
a_{mxcurl}	Last maximum of local angle along ray	amxcurl	N/A
angu	Maximum tangent angle from source to terrain peaks.	angu	N/A
ant_{fac}	Antenna pattern parameter (depends on i_{ptrn} and μ_{bwr})	afac	pattern

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
ant_{k_o}	Transmitting antenna height times the free space wavenumber k_o	antko	N/A
ant_{ref}	Transmitting antenna height relative to the ground height at range 0.	antref	miscvar
a_r	Complex coefficient of partial linear solution to homogeneous equation	ar	N/A
a_{rx}	Partial linear solution to homogeneous equation	arx	N/A
a_s	Starting launch angle in radians	as	N/A
a_{sl}	Last starting launch angle value	asl	N/A
b	Argument of internal function in TRACEA SU	b	N/A
b_r	Complex coefficient of partial linear solution to homogeneous equation	br	N/A
b_{rx}	Partial linear solution to homogeneous equation	brx	N/A
c	Argument of internal function in TRACEA SU	c	N/A
c_o	Speed of light in m/s	c0	N/A
C_1	Coefficient used in vertical polarization calculations	c1	impedance
C_{1C}	summation argument in determining a_r	c1c	N/A
C_{1M}	Constant for each calculated α_v used in C_1 calculation	c1m	impedance
C_2	Coefficient used in vertical polarization calculations	c2	impedance
C_{2C}	summation argument in determining b_r	c2c	N/A
C_{2M}	Constant for each calculated α_v used in C_2 calculation	c2m	impedance
c_{ak}	Double precision internal variable in PHASE1 SU	cak	N/A
cd	$R_{AV}(ii)$ or $-R_{AV}(ii)$ depending on power index	cd	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	**Description**	**FORTRAN source code name**	**FORTRAN COMMON BLOCK name**
cnst	Constant equals $\Delta p / k_o$	cnst	miscvar
con	$10^{-6} k_o$	con	pevar
Δp	Mesh size in angle- (or p-) space	delp	miscvar
$\Delta\Theta$	Difference between mesh points in p-space	dtheta	N/A
D_{term}	Field due a real point source at height $h_{transmitter}$	dterm	N/A
$\partial M(j_l)/\partial h$	Gradient of first profile at j_l in M-units/meter	dmdh(jl)	trvar
Δx_{out}	Output range step	drout	rhstps
Δx_{k_0}	Internal variable in PHASE1 SU	drfk	N/A
Δx_{PE}	PE range step	dr	rhstps
Δx_{PE2}	½ PE range step	dr2	rhstps
Δz_{out}	Output height increment	dzout	rhstps
Δz_{PE}	Bin width in z space	delz	pevar
Δz_{PE2}	2. * Δz_{PE}	dz2	pevar
ef	Error flag structure for external implementation constants	ef	N/A
ef.lerr16	Element of user-provided error flag structure *ef* that will trap on certain errors if set to .TRUE	ef.lerr6	N/A
ef.lerr12	Element of user-provided error flag structure that will trap on certain errors if set to .TRUE	ef.lerr12	N/A
elerr12	Element of user-provided error flag structure *ef* that will trap on certain errors if set to .TRUE	elerr12	N/A

Table7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
envpr	Complex refractivity profile array interpolated every Δz_{PE} in height	envpr	arrays
ε_r	Relative permittivity	epsilon	N/A
$f(\alpha)$	Antenna pattern factor for specified elevation angle	patfac	N/A
fac_D	Antenna pattern factor for direct ray angle	facd	N/A
fac_R	Antenna pattern factor for reflected ray angle	facr	N/A
farray	Field array to be propagated one range step in free space	farray	N/A
f_{MHz}	Frequency	sv.freq	N/A
f_{fac}	Propagation factor at specified height	getpfac	N/A
FILT	Cosine-tapered (Tukey) filter array	filt	arrays
f_{norm}	Normalization factor	fnorm	miscvar
f_r	Interpolation fraction	fr	N/A
frac	Fractional distance from pl_1 to pl_2	frac	N/A
frac	Fraction variable used internally in PROFREF SU	frac	N/A
frsp	Complex free space propagator term array	frsp	arrays
$fspace_{x_{out}}$	Free space loss at x_{out}	fslrout	N/A
fter	Logical flag indicating if performing terrain case	fter	miscvar
fv	Fraction used in the interpolation of profiles in the REFINTER SU	fv	N/A
grad	Gradient of current refractivity layer	grad	N/A
$grad_j$	Gradient of j^{th} refractivity layer	grad	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	**Description**	**FORTRAN source code name**	**FORTRAN COMMON BLOCK name**
h_0	Height of the ray before the trace step	h0	N/A
h_1	Height of the ray after the trace step	h1	N/A
h_{dif}	Height difference between last two differing height levels in each refractivity profile	hdif	N/A
$h_{l\arg e}$	Maximum height at which the refractivity profile is extrapolated	hlarge	N/A
h_{ref}	Heights of refractivity profile with respect to local ground height	href	profwref
htdum	Dummy array containing height values for current (horizontally interpolated) profile	htdum	N/A
ht	PE mesh height array of size n_{fft}	ht	parinit
ht_{lim}	Maximum calculation height with respect to y_{minter}	htlim	miscvar
$h_{transmitter}$	Transmitting antenna height above the local ground	sv.antht	N/A
i	Imaginary i = complex (0,1)	qi	misvar
i	Index used in the determination of *rfac1* and *rfac2*	i	N/A
i	Index used in PROFREF	i	N/A
i	Index used in the determination of the field *U*(*i*) in XYINIT SU	i	N/A
i	Index used in the filtering of the *frsp* array in PHASE1 SU	i	N/A
I	Index used in determination of *frsp*(*I*) in PHASE1 SU	I	N/A
I	Index used in the determination of *profint* in INTPROF SU	I	N/A
ibsml	Flag indicating if y_{ref} is below mean sea level in PROFREF SU	ibsml	N/A
i_c	Counter for the array $y_{\lim}$	ic	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
i_{dn}	Index used to increment or decrement the initial launch angle a_s	idn	N/A
ierror	Flag indicating if last profile entered was at a range less than vr_{max}	ierror	N/A
i_{flag}	Index indicating whether the refractivity profile is to be referenced to y_{minter} or to local ground height above y_{minter}	iflag	N/A
i_g	Counter indicating current ground type being modeled	ig	impedance
i_{hu}	Range index at which the traced ray has reached the maximum calculation height	ihu	N/A
i_{m1}	Index used in loops for counter decrements, i.e., $i_{m1} = i - 1$	im1	N/A
i_{p1}	Index used to set the minimum propagation factor for *rfac*1	ip1	N/A
i_{p1}	Index used in loops for counter increments, i.e. $i_{p1} = i + 1$	ip1	N/A
i_{p2}	Index used to set the minimum propagation factor for *rfac2*	ip2	N/A
i_{ptrn}	Type of antenna pattern desired	sv.ipat	N/A
i_{ptrn}	Antenna pattern type	iptrn	N/A
i_s	Counter for current profile	is	parinit
i_{set}	Flag to test whether or not to stop loop used to determine the launch angle	iset	N/A
i_{zg}	Number of height points corresponding to local ground height at current output range x_{out}	izg	N/A
j	Index of the last refractivity profile	j	N/A
j	Index used in the determination of *profint* in INTPROF SU	j	N/A
j_{end}	Index at which valid loss values in *mloss* end	jend	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
j_k	Loop index used in PROFREF SU	jk	N/A
j_l	Index of current refractivity layer ray tracing through	jl	N/A
j_{ls}	Index of the refractivity array at which the antenna height is located	jls	trvar
j_s	Index used in PROFREF SU	js	N/A
j_{start}	Index at which valid loss values in *mloss* begin	jstart	N/A
k	Index used in determination of *mloss*	k	N/A
k	Index used in PROFREF	k	N/A
k_o	Free-space wave number $= 2\pi / \lambda$	fko	pevar
k_{bin}	Number of bins to be shifted	kbin	N/A
k_p	Counter for terrain profile	kp	N/A
k_t	Counter for terrain profile	kt	N/A
λ	Wave length	wl	pevar
ln_{fft}	Power of 2 transform size, i.e. $n_{fft} = 2**ln_{fft}$	ln	pevar
loop	Flag used to test whether or not to stop a loop	loop	N/A
lvlep	Number of height/refractivity levels in profile	lvlep	parinit
lvlm1	Last user-specified level in refractivity profile	lvlm1	N/A
lvlm2	Second-to-last user specified level in refractivity profile (i.e., *lvlm1*-1)	lvlm2	N/A
M_o	Modified refractivity at ant_{ref}	rmatht	N/A
M_a	Minimum modified refractivity above ant_{ref}	rmina	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
M_b	Minimum modified refractivity below ant_{ref}	rminb	N/A
maxn4	*maxpts* divided by 4; specifies the length of the filter array	maxn4	N/A
maxpts	Maximum size of arrays for the real and imaginary fields	maxpts	N/A
mloss	Loss values	mloss	N/A
μ_{0r}	Antenna pattern elevation angle	elv	pattern
μ_{bwr}	Antenna vertical beam width	bw	pattern
μ_{max}	Limiting angle for SIN(X)/X and generic height finder antenna pattern factors	umax	pattern
mxlvls	Maximum number of height/M-unit levels	mxlvls	N/A
mxnfft	Maximum power of 2 for transform size	mxnfft	N/A
mxrout	Maximum number of output range points	mxrout	N/A
mxter	Maximum number of height/range points allowed for terrain profile	mxter	N/A
mxzout	Maximum number of output height points	mxzout	N/A
newl	New level index used internally in PROFREF SU	newl	N/A
n_{fft}	Transform size	n	pevar
nm1	$n_{fft} - 1$	nm1	pevar
n_{lvl}	Number of levels in new profile	nlvl	profwref
n_p	Final number of refractivity profiles	np	N/A
$n_{3/4}$	¾ of n_{fft}	n34	pevar
nvrout	Number of output range points	nvrout	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
p_{angle}	User specified maximum propagation angle	prang	N/A
pfacmin	Minimum propagation factor	pfacmin	N/A
π	3.1415926	pi	N/A
pl_1	Point one	pl1	N/A
pl_2	Point two	pl2	N/A
plcnst	Constant used in determining propagation loss $\left(20\log\left\{2k_o\right\}\right)$	plcnst	miscvar
pow	Interpolated amplitude of the field at the receiver height z_r	pow	N/A
profint	Profile interpolated to every Δz_{PE} in height	profint	parinit
range	Range for profile interpolation	range	N/A
$R_{AV}(ii)$	Array of R_T to the iith power (e.g., R_T^i)	rav	impedance
refcoef	Complex reflection coefficient	refcoef	N/A
refdum	Dummy array containing refractivity values for current (horizontally interpolated) profile	refdum	N/A
ref_{ref}	Refractivity array	refref	profwref
rf	Refractivity structure for external environmental data elements	rf	N/A
rfac1	Array of propagation factors at valid output height points for range x_{last}	rfac1	N/A
rfac2	Array of propagation factors at valid output height points for range x_{PE}	rfac2	N/A
rf	Refractivity structure for external environmental data elements	rf	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
rf.hmsl	2-dimensional array containing heights with respect to mean sea level of each profile. Array format must be *hmsl(i,j)* = height of i^{th} level of j^{th} profile. j=1 for range-independent cases	rfhmsl	N/A
rf.lvlep	Number of levels in refractivity profile	rf.lvlep	N/A
rf.nprof	Number of profiles	rf.nprof	N/A
rf.refmsl	2-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be *refmsl(i,j)* = M-unit at i^{th} level of j^{th} profile. j=1 for range-independent cases	rf.refmsl	N/A
rf.rngprof	Ranges of each profile. *rngprof(i)* = range of *i*th profile	rf.rngprof	N/A
r_{hor}	Radar horizon range based on transmitter height $h_{transmitter}$ and 0 receiver height	rhor	N/A
R_K	Coefficient used in C_1 and C_2 calculations	rk	impedance
$r_{l\,arge}$	Maximum range at which the refractivity profile is extrapolated	rlarge	N/A
R_{ng}	Complex refractive index	rng	impedance
R_{ng2}	Complex refractive index squared	rng2	impedance
r_{ref}	Range at which the ray is reflected	rref	N/A
R_T	Complex root of quadratic equation for mixed transform method based on Kuttler's formulation	root	impedance
R_{term}	Field due to an image point source at the height $-h_{transmitter}$	rterm	N/A
rv1	Range of the last refractivity profile	rv1	N/A
rv2	Range of the next refractivity profile	rv2	parinit

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
scd	Terrain slope difference	scd	N/A
s_{gain}	Normalization factor	sgain	N/A
σ	Conductivity	sigma	N/A
$\mathrm{SIN}\left(\mu_{0r}\right)$	Sine of antenna elevation angle	pelev	pattern
$\mathrm{SIN}\left(\mu_{bwr}\right)$	Sine of antenna vertical beam width	sbw	pattern
$\mathrm{SIN}\left(\alpha\right)$	Sine of specified elevation angle	sang	N/A
slope	Current slope of terrain segment	slope	N/A
slp	Slope of each segment of terrain	slp	miscvar
sum1	summation term in determining a_r	sum1	N/A
sum2	summation term in determining b_r	sum2	N/A
sv	System structure for external system data elements	sv	N/A
sv.bwidth	Half power (3 dB) antenna pattern beamwidth	sv.bwidth	N/A
sv.elev	Antenna pattern elevation angle	sv.elev	N/A
sv.polar	Character string indicating polarization	sv.polar	N/A
Θ_{15}	15 degrees in radians	deg15	N/A
Θ_{luanch}	Launch angle	thetalaunch	trvar
Θ_{max}	Maximum propagation angle in PE calculations	thetamax	miscvar
tr	Terrain structure for external terrain data elements	tr	N/A
tr.dielec	2-dimensional array containing the relative permittivity and conductivity for user defined terrain	tr.dielec	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
$tr.i_{gr}$	Number of different ground types specified	tr.igr	N/A
tr.igrnd	Type of ground composition for given terrain profile	tr.igrnd	N/A
tr.itp	Number of points in profile	tr.itp	N/A
tr.rgrnd	Ranges at which the ground types apply	tr.rgrnd	N/A
tr.terx	Range points of terrain profile	tr.terx	N/A
tr.tery(*i*)	Height points of the terrain profile	tr.tery(i)	N/A
U	Complex PE field	u	arrays
U_i	$U(i)$	ui	N/A
U_{last}	Complex field at previous PE range	ulst	arrays
U_{nmi}	$U(n-i)$	unmi	N/A
vnp	*INPUTVAR* structure for external implementation constants	vnp	N/A
vnp.nrout	Integer number of output range points desired	vnp.nrout	N/A
vnp.nzout	Integer number of output height points desired	vnp.nzout	N/A
vnp.propang	Maximum problem angle in degrees	vnp.propang	N/A
vr_{max}	Maximum range	vrmax	N/A
w	Difference equation of complex PE field array (used only for vertical polarization)	w	N/A
X	Real part of field	x	N/A
x_0	Range of ray before trace step	r0	N/A
x_1	Range of ray after trace step	r1	N/A

Table 7-2 Variable name cross reference (con't)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
x_{cur}	Current PE range	r	N/A
x_i	Range of i^{th} terrain point	xi	N/A
x_{last}	Last PE range	rlast	N/A
x_{lim}	90% of the maximum range, x_{max}, used for ray tracing	rlim	trvar
x_{log}	10 times the logarithm of x_{PE}	rlog	N/A
x_{loglst}	10 times the logarithm of x_{last}	rloglst	N/A
x_{m1}	Range of $(i\text{-}1)^{th}$ terrain point	xm1	N/A
x_{max}	Maximum output range	rmax	N/A
x_{max}	Maximum output range	vnp.rmax	N/A
x_{mid}	Range at which interpolation for range-dependent refractivity profiles is performed	rmid	N/A
x_o	Current output range at which to store the height of traced ray in y_{lim}	ro	N/A
x_{out}	Output range	rout	N/A
x_{PE}	PE range	r	N/A
x_{PE}	Range at which valid loss values will begin to be calculated	rpe	miscvar
xx	Fractional range at which to interpolate the propagation factor	xx	N/A
Y	Complex part of field	x	N/A
y_{ch}	Height of the terrain at the current PE range relative to y_{mref}	ych	N/A
y_{ct}	Height of the terrain at the current PE range relative to y_{minter}	yct	N/A
y_{cur}	Height of ground at current PE range	ycur	htvar

Table 7-2 Variable name cross reference (cont'd)

SDD variable name	Description	FORTRAN source code name	FORTRAN COMMON BLOCK name
y_{curm}	Height of ground midway between last and current PE range	ycurm	htvar
y_i	Height of i^{th} terrain point	yi	N/A
y_{last}	Height of ground at last PE range	ylast	htvar
y_{lh}	Height of the terrain at the last PE range relative to y_{mref}	ylh	N/A
y_{lim}	Height at each output range at which the last valid loss value exists	hlim	miscvar
y_{lt}	Height of the terrain at the last PE range relative to y_{minter}	ylt	N/A
y_m	Particular solution of Kuttler's difference equation	ym	N/A
ym	Particular solution of difference equation (used in intermediate calculations only for vertical polarization)	ym	N/A
y_{m1}	Height of $(i\text{-}1)^{th}$ terrain point	ym1	N/A
y_{max}	Maximum output height	vnp.hmax	N/A
y_{min}	Minimum output height	vnp.hmin	N/A
y_{minter}	Reference height for internal calculations of the field U (minimum height of terrain profile)	hminter	N/A
y_{mref}	Height relative to y_{minter}	hmref	miscvar
y_{ref}	Reference height at current range step	yref	N/A
y_{termax}	Maximum height of terrain profile	htermax	N/A
z_{lim}	Height limit for ray trace	zlim	trvar
z_{limit}	Maximum height region where PE solution is valid	zlim	trvar

Table 7-2 Variable name cross reference (con't)

SDD variable name	**Description**	**FORTRAN source code name**	**FORTRAN COMMON BLOCK name**
z_{max}	Total height of the FFT/PE calculation domain	zmax	pevar
z_{int}	Interpolated ground height at current range x_{out}	zint	N/A
z_{out}	Array containing all output height points	zout	fhstps
z_r	Receiver height	height	N/A

8. APPENDIX A: Fortran SOURCE CODE FOR TPEM CSCI

```
c ***************** THIS FILE CONTAINS TPEM MODEL SUBROUTINES ***************

c Author: Amalia E. Barrios
c         NCCOSC RDT&E DIV D883
c         49170 Propagation Path
c         San Diego, CA  92152-7385
c         e-mail: barrios@nosc.mil
c         phone: (619) 553-1429
c         fax: (619) 553-1417

c Summary: These routines model tropospheric radiowave propagation over
c          variable terrain and calculates propagation loss vs. height and
c          range. Propagation loss is displayed in dB contours on a height vs.
c          range plot. TPEM is based on the split-step Fourier PE method and
c          was originally developed from an early PE model called PEPC, written
c          by Fred Tappert.  Propagation loss over variable terrain is modeled
c          by shifting the field an appropriate number of bin widths correspond-
c          ing to the height of the ground.  The field is determined using the
c          smooth earth PE method.

c ****************************************************************************

c Variables in small letters in parameter lists are variables that are input
c or passed to called subroutines.  Variables in CAPS in parameter lists are
c returned from the called subroutines.

c ****************************************************************************

c Main Glossary of Common Blocks used in all subroutines:

c ARRAYS:
c   ENVPR() = Complex array containing refractivity exponential term.
c              i.e. ENVPR() = exp[i * dr * k * 1e-6 * M(z) ]
c   FILT() = Cosine-tapered (Tukey) filter array.
c   FRSP() = Complex array containing free-space propagator exponential term.
c             i.e., FRSP() = exp[-i * dr * (k - sqrt(k**2 - p**2)) ].
c   U() = Complex array containing field solution at current PE range.
c   ULST() = Complex array containg field solution at previous PE range.

c HTVAR:
c   YLAST = Height of ground in meters at the last PE range.
c   YCUR = Height of ground in meters at the current PE range.
c   YCURM = Height of ground in meters midway between last and current PE range
c           step. For use when shifting profiles to be relative to the local
c           ground height.

c IMPEDANCE:
c   ALPHAV = Vertical polarization impedance term = i*fko/rng.
c   C1 = Coefficient used in vertical polarization calculations.
c   C2 = Coefficient used in vertical polarization calculations.
c   C1M = Constant for each calculated ALPHAV - used in C1 calculation.
c   C2M = Constant for each calculated ALPHAV - used in C2 calculation.
c   IG = Counter indicating current ground type being modeled.
c   RAV() = Array of ROOT to the i'th power, i.e. RAV(I) = ROOT**I.
c   RK = Coefficient used in C1 and C2 calculations.
c   RNG = Complex refractive index.
c   RNG2 = Complex refractive index squared.
c   ROOT = Complex root of quadratic equation for mixed transform method
c          based on Kuttler's formulation.

c MISCVAR:
c   ANTREF = Transmitting antenna height relative to the reference
c            height HMINTER.
```

```
c   CNST = Used in calculating FRSP() in routine PHASE1.
c          CNST = DELP/FKO.
c   DELP = Mesh size in angle- (or p-) space.
c   FNORM = Normalization factor used for DFT.
c   FTER = Logical flag - .TRUE.=terrain case, .FALSE.=smooth surface case
c   HLIM() = Array containing height at each output range at which the
c            last valid loss value exists.
c   HMREF = Height relative to HMINTER.  Determined from user-provided
c           minimum height VNP.HMIN. That is, if VNP.HMIN is minimum
c           height input by user with respect to mean sea level,
c           and HMINTER is internally considered the new origin,
c           then HMREF = VNP.HMIN - HMINTER.
c   HTLIM = Maximum desired calculation height with respect to HMINTER,
c           i.e., HTLIM = VNP.HMAX-HMINTER.
c   PLCNST = Constant used in determining propagation loss.
c            PLCNST = 20log(2*FKO).
c   QI = Imaginary i -> complex(0,1).
c   RPE = Range at which valid loss values will begin to be calculated.
c   THETAMAX = Maximum propagation angle in PE calculations.
c   SLP() = Slope of each segment of terrain.


c PARINIT:
c   HT() = PE mesh height array of size N.
c   HTDUM() = Dummy array containing height values for current (horizontally
c             interpolated) profile.
c   IS = Counter for current profile (for range-dependent cases).
c   LVLEP = Number of height/refractivity levels in profile.
c   PROFINT() = M-unit profile interpolated to every DELZ in height.
c   REFDUM() = Dummy array containing M-unit values for current (horizontally
c              interpolated) profile.
c   RV2 = Range of the next refractivity profile (for range-dependent cases).

c PATTERN:
c   AFAC = Constant used in determining antenna pattern factors.
c          AFAC = 1.39157 / sin( bw / 2 ) for SIN(X)/X and height-finder.
c          AFAC = (.5*ln(2))/(sin(bw/2))**2 for GAUSSIAN.
c   BW = Antenna pattern beamwidth in radians.
c   ELV = Antenna pattern elevation angle in radians.
c   PELEV = Sine of elevation angle.
c   SBW = Sine of the beamwidth.
c   UMAX = Limiting angle used in 30 dB cut-off point for SIN(X)/X and
c          generic height-finder antenna pattern factors.

c PEVAR:
c   CON = 1.e-6 * FKO; Constant used in calculation of ENVPR().
c   DELZ = Bin width in z-space = WL / (2*sin(THETAMAX)).
c   DZ2 = 2. * DELZ.
c   FKO = Free-space wavenumber = (2*pi) / WL
c   LN = Power of 2 transform size, i.e. N = 2**LN.
c   N = Transform size.
c   N34 = 3/4 * N.
c   NM1 = N-1.
c   WL = Wavelength in meters.
c   ZMAX = Maximum height of PE calculation domain = N * DELZ.

c PROFWREF:
c   HREF() = Heights of refractivity profile with respect to local ground
c            height.
c   NLVL = Number of levels in new profile.
c   REFREF() = Corresponding refractivity array for HREF().

c RHSTPS:
c   DR = PE range step in meters.
c   DR2 = 1/2 PE range step in meters.
c   DROUT = Output range step in meters.
c   DZOUT = Output height increment in meters.
```

```
c   ZOUT() = Array containing all output height points.


c TRVAR:
c   DMDH() = Gradients of first profile in M-units/meters.
c   JLS = Index of refractivity array at which antenna height is located.
c   RLIM = 90% of maximum range RMAX - used for ray tracing.
c   THETALAUNCH = Angle in radians of launch angle for which, when traced,
c                 height of the ray at each output range step is stored.
c   ZLIM = Height limit for ray trace.
```

8.1 Subroutine PEINIT

```
c ************************** SUBROUTINE PEINIT ******************************

c Module Name: PEINIT

c Module Security Classification: UNCLASSIFIED

c Purpose:  Initializes all variables used in TPEM subroutines for PE calcula-
c           tions.

c Version Number: 1.5

c INPUTS:
c    Argument list: EF(errorflag) structure, RF(refractivity) structure,
c                   SV(systemvar) structure, TR(terrain) structure,
c                   VNP(inputvar) structure

c OUTPUTS:
c    Argument list: HMINTER, IERROR, ROUT
c    Common: AFAC, ALPHAV, ANTREF, BW, C1, C2, C1M, C2M, CNST, CON,
c            DELP, DELZ, DMDH(), DR, DR2, DROUT, DZ2, DZOUT, ELV, ENVPR(),
c            FILT(), FKO, FNORM, FRSP(), FTER, HLIM(), HMREF, HT(), HTDUM(),
c            HTLIM, IG, IS, JLS, LN, LVLEP, N, N34, NM1, PELEV, PLCNST,
c            PROFINT(), QI, RAV(), REFDUM(), RPE, RK, RLIM, RNG, RNG2, ROOT,
c            RV2, SBW, SLP(), THETALAUNCH, THETAMAX, U(), UMAX, WL, YCUR,
c            YCURM, YLAST, ZLIM, ZMAX, ZOUT()
c

c FILES INCLUDED: FFTSIZ.INC, TPEM.INC

c CALLING ROUTINES: MAIN DRIVER PROGRAM or TESS CSCI

c ROUTINES CALLED: DIEINIT, FFT, GETFFTSZ, INTPROF, PHASE1, PHASE2, PROFREF,
c                  REFINIT, TRACEA, TRACEH, XYINIT

c GLOSSARY:
c    EF = Error flag structure for external implementation constants.
c       EF.LERR6 =  Logical flag that allows for greater flexibility in
c                   allowing error -6 to be bypassed.  If set to .TRUE.
c                   then trapping for this erroroccurs, otherwise it can
c                   be totally ignored by main driver program. (Within the
c                   TPEM program it is handled as a warning).  If this
c                   error is bypassed (EF.LERR6 = .FALSE.) terrain profile
c                   is extended to RMAX with same elevation height of last
c                   valid terrain profile point.
c       EF.LERR12 = Same as EF.LERR6 - allows for trapping of this error.
c                   If LERR12 = .FALSE., then (for range-dependent case)
c                   if range of last refractivity profile entered is less
c                   than RMAX, the environment is treated as homogeneous
c                   from the last profile entered to RMAX.
c    RF = Refractivity structure for external environmental data elements.
c       RF.LVLEP = Number of levels in refractivity profile (for range
c                  dependent case all profiles must have same number of
c                  levels).
c       RF.REFMSL(,) = 2-dimensional array containing refractivity with
c                      respect to mean sea level of each profile.  Array
c                      format must be REFMSL(I,J) = M-unit value at Ith
c                      level of Jth profile.  J = 1 for range-independent
c                      cases.
c       RF.HMSL(,) = 2-dimensional array containing heights in meters with
c                    respect to mean sea level of each profile.  Array format
c                    must be HMSL(I,J) = height of Ith level of Jth profile.
c                    J = 1 for range-independent cases.
c       RF.RNGPROF() = Ranges of each profile in meters, i.e., RNGPROF(I) =
```

```
c                      range of Ith profile.  RNGPROF(1) should always be
c                      equal to 0.
c        RF.NPROF = Number of profiles.  Equals 1 for range-independent cases.
c     SV = System structure for external system data elements.
c        SV.FREQ = Frequency in MHz.
c        SV.ANTHT = Transmitting antenna height above local ground in meters.
c        SV.BWIDTH = Half-power (3 dB) antenna pattern beamwidth in degrees
c                   (.5 to 45.).
c        SV.ELEV = Antenna pattern elevation angle in degrees. (-10 to 10).
c        SV.POLAR = 1-character string indicating polarization.  H-horizontal,
c                  V-vertical
c        SV.IPAT = Integer value indicating type of antenna pattern desired.
c                 IPAT = 0 -> omni
c                 IPAT = 1 -> gaussian
c                 IPAT = 2 -> sinc x
c                 IPAT = 3 -> csc**2 x
c                 IPAT = 4 -> generic height-finder
c     TR = Terrain structure for external terrain data elements.
c        TR.TERX() = Range points of terrain profile in meters.
c        TR.TERY() = Reight points of terrain profile in meters.
c        TR.ITP = Number of height/range pairs in profile.
c        TR.IGR = Number of different ground types specified.
c        TR.IGRND() = Type of ground composition for given terrain profile -
c                    can vary with range.  Different ground types are:
c                    0 = sea water, 1 = fresh water, 2 = wet ground,
c                    3 = medium dry ground, 4 = very dry ground,
c                    5 = user defined (in which case, values of relative
c                    permittivity and conductivity must be given).
c        TR.RGRND() = Ranges at which the ground types apply.
c        TR.DIELEC(,) = 2-dimensional array containing the relative
c                      permittivity and conductivity; DIELEC(1,i) and
c                      DIELEC(2,i), respectively. Only needs to be specified
c                      if using IGRND(i) = 5, otherwise, TPEM will
c                      calculate based on frequency and ground types 0-4.
c     VNP = Inputvar structure for external implementation constants.
c        VNP.HMAX = Maximum output height with respect to m.s.l. in meters.
c        VNP.HMIN = Minimum output height with respect to m.s.l. in meters.
c        VNP.RMAX = Maximum output range in meters.
c        VNP.NZOUT = Integer number of output height points desired.
c        VNP.NROUT = Integer number of output range points desired.
c        VNP.PROPANG = Maximum problem (propagation) angle in degrees
c                     desired for solution. If set to 0., then TPEM will
c                     determine it's own.
c     HMINTER = Height of the minimum elevation of terrain profile.  This
c               will be used to adjust entire terrain profile so subsequent
c               loss values returned will be referenced to this height.
c     IERROR = Integer value that is returned if any errors exist in input data:
c              -6 : Last range in terrain profile is less than VNP.RMAX. (Will
c                   only return this error if error flag EF.LERR6 is set to
c                   .TRUE.).
c              -8 : VNP.HMAX is less than maximum height of terrain profile.
c             -12 : Range of last refractivity profile entered (for range depen-
c                   dent case) is less than RMAX. (This is returned from subrou-
c                   tine REFINIT).  Will only return this error if error flag
c                   EF.LERR12 is set to .TRUE.).
c             -14 : Last gradient in any refractivity profile entered is
c                   negative.
c                   (This is returned from REFINIT).
c             -17 : Range points of terrain profile is not increasing.
c             -18 : First range point is not 0.
c             -42 : Minimum height input by user (VNP.HMIN) is greater then
c                   maximum height (VNP.HMAX).
c     ROUT = Output range point (meters) - initialized in this routine

c     For output variables in common blocks, look in main glossary for
c     associated common blocks for variable definitions.
```

```
c     Common blocks: variables
c        ARRAYS: ENVPR(), FILT(), FRSP(), U()
c        PEVAR: CON, DELZ, DZ2, FKO, LN, N, N34, NM1, WL, ZMAX,
c        RHSTPS: DR, DR2, DROUT, DZOUT, ZOUT()
c        MISCVAR: ANTREF, CNST, DELP, FNORM, FTER, HLIM(), HMREF, HTLIM,
c                 PLCNST, QI, RPE, SLP(), THETAMAX
c        PATTERN: AFAC, BW, ELV, PELEV, SBW, UMAX
c        PARINIT:  HT(), HTDUM(), IS, LVLEP, PROFINT(), REFDUM(), RV2
c        HTVAR: YCUR, YCURM, YLAST
c        TRVAR: DMDH(), JLS, RLIM, THETALAUNCH, ZLIM
c        IMPEDANCE: ALPHAV, C1, C2, C1M, C2M, IG, RAV(), RK, RNG, RNG2, ROOT

c     Local Variables:
c        ACRIT = Critical angle, i.e., angle above which no rays are trapped
c                for ducting environment.
c        ANGLE = Tangent ray angle from source to each terrain peak along
c                profile.
c        ANGU = Maximum tangent ray angle from souce to terrain peak along
c               terrain profile path
c        BUGFIX = Dummy variable used to push/move bits in EOF statement.
C                 This is a known bug in MS Fortran Powerstation 1.0 and is
c                 not necessary for other flavors of Fortran.
c        C0 = Speed of light in m/s
c        C1C = Intermediate complex number used to calculate C1
c        C2C = Intermediate complex number used to calculate C2
c        CN75 = 4 * pi / N
c        CX = RAV(I) if I is even or -RAV(I) if I is odd
c        DX1 = Difference between current and previous "unprocessed" range
c              points (i.e.,XI-XM1)
c        DX2 = Difference between current and next "unprocessed" range points
c              (i.e., XP1-XI)
c        GRAD = Gradient of current refractivity layer in profile
c        H1 = (I)th height value in first refractivity profile
c        H2 = (I+1)th height value in first refractivity profile
c        HDEG = 1/2 degree in radians
c        HTERMAX = Maximum height in meters along terrain profile.
c        ISCR = Unit number for scratch file - used for temporary storage of
c               terrain profile points for processing.
c        LFLAG = Logical flag indicating if end-of-file has been reached in the
c                scratch file for reading back processed terrain points.
c        LOPEN = Logical flag returned in INQUIRE statement that checks if a
c                file is already attached to unit ISCR.
c        NO4 = N/4
c        PRANG = User-specified propagation angle in radians.
c        RADC = PI/180 -> used for converting angles given in degrees to
c               radians.
c        RHOR = Radar horizon range based on transmitter height ANTREF and
c               0 receiver height.
c        RKM = Maximum range in km.
c        RLLIM = Various maximum range step limits based on geometry and
c                whether terrain or smooth surface case is being performed
c        RM1 = (I)th M-unit value in first refractivity profile
c        RM2 = (I+1)th M-unit value in first refractivity profile
c        RMATHT = M-unit value at transmitter height ANTREF
c        RMINA = Minimum M-unit value above transmitter height
c        RMINB = Minimum M-unit value for all heights below transmitter height
c        SCD = Difference in previous and next "unprocessed" slope segments
c              (i.e., SCD = second derivative, d^2y/dx^2 )
c        SDEG10 = Sine of 10 degrees.
c        SDEG15 = Sine of 15 degrees.
c        SL1 = Slope of previous "unprocessed" terrain segment
c        SL2 = Slope of next "unprocessed" terrain segment
c        SLOPE = Slope of current terrain segment (after processing)
c        STHETAMAX = Sine of THETAMAX
c        THETAFRAC = Fractional number relating THETALAUNCH to THETAMAX
c        UI = (I)th value of complex PE field
c        UNMI = (N-I)th value of complex PE field
```

```
c       X1 = Range of Ith terrain point in meters (after processing)
c       X2 = Range of (I+1)th terrain point in meters (after processing)
c       XI = Range of "unprocessed" (I)th terrain point in meters
c       XDIF = Range difference between current and next terrain points after
c              processing (i.e., X2-X1)
c       XM1 = Range of "unprocessed" (I-1)th terrain point in meters
c       XP1 = Range of "unprocessed" (I+1)th terrain point in meters
c       Y1 = Height of Ith terrain point in meters (after processing)
c       Y2 = Height of (I+1)th terrain point in meters (after processing)
c       YDIF = Height difference between current and next terrain points after
c              processing (i.e., Y2-Y1)
c       YI = Height of "unprocessed" (I)th terrain point in meters
c       YM1 = Height of "unprocessed" (I-1)th terrain point in meters
c       YP1 = Height of "unprocessed" (I+1)th terrain point in meters

      subroutine peinit( ef, vnp, rf, sv, tr, HMINTER, ROUT, IERROR )

      include 'tpem.inc'

      common / arrays / u(0:maxpts), filt(0:maxn4), frsp(0:maxpts),
     +                  envpr(0:maxpts), ulst(0:maxpts)
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / rhstps / dr, drout, dzout, dr2, zout(mxzout)
      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
      common / pattern / pelev, afac, bw, elv, umax, sbw
      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
     +                   profint(0:maxpts), ht(0:maxpts), is, lvlep
      common / htvar / ylast, ycur, ycurm
      common / trvar / dmdh(mxlvls), zlim, jls, thetalaunch, rlim
      common / impedance / alphav, rav(0:maxpts), rng, rng2, c1, c2,
     +                     rk, c1m, c2m, ig, root

      record / errorflag / ef
      record / inputvar / vnp
      record / refractivity / rf
      record / systemvar / sv
      record / terrain / tr

      complex u, frsp, envpr, ulst, qi, alphav, rav, rng, rng2, c1c
      complex c2c, c2m, rk, c1m, c1, c2, root, ui, unmi, cx

      logical fter, lopen, lflag

      data radc / 1.74533e-2 /        !degree to radian conversion factor
      data iscr / 20 /                ! Unit number for scratch file
      data c0 / 299.79245 /           !speed of light x 1e.-6 m/s
      data sdeg10 / .173648177 /      ! Sine of 10 degrees
      data sdeg15 / .258819045 /      ! Sine of 15 degrees
      data hdeg / 8.726646e-3 /       ! 1/2 degree

      ierror = 0
      fter = .false.
      thetamax = 0.
      hminter = 0.
      angu = 0.
      antref = sv.antht

c Put lower limit on HMAX and RMAX

      vnp.rmax = amax1( vnp.rmax, 5000. ) !Set max. range to no less than 5 km.
      vnp.hmax = amax1( vnp.hmax, 100. )  !Set max. height to no less than 100 m.
      if( vnp.hmin .ge. vnp.hmax ) then
         ierror = -42
         return
      end if
```

```
      vnp.hmin = amin1( vnp.hmin, vnp.hmax-100. )

      dzout = (vnp.hmax-vnp.hmin) / float( vnp.nzout )
      drout = vnp.rmax / float( vnp.nrout )

c Setup output height array

      do i = 1, vnp.nzout
         zout(i) = vnp.hmin + float(i) * dzout
      end do

      WL = c0 / sv.freq
      FKo = 2. * pi / WL
      con = 1.e-6 * fko
      qi = cmplx( 0., 1. )

c  Calculate constants used to determine antenna pattern factor
c SV.IPAT = 0 -> omni
c SV.IPAT = 1 -> gaussian
c SV.IPAT = 2 -> sinc x
c SV.IPAT = 3 -> csc**2 x
c SV.IPAT = 4 -> generic height-finder

      sv.bwidth = amax1( sv.bwidth, .5 )    !Don't let vertical beamwidth go
      sv.bwidth = amin1( sv.bwidth, 45. )   !outside .5 to 45 degree limit.

      sv.elev = amax1( sv.elev, -10. ) !Don't let elevation angle go
      sv.elev = amin1( sv.elev, 10. )  !outside -10 to 10 degree limit.

      bw = sv.bwidth * radc
      elv = sv.elev * radc
      bw2 = .5 * bw
      if( sv.ipat .eq. 1 ) then
         afac = .34657359 / (sin( bw2 ))**2
         pelev = sin( elv )
      elseif( sv.ipat .eq. 3 ) then
         sbw = sin( bw )
      elseif( sv.ipat .ne. 0 ) then
         afac = 1.39157 / sin( bw2 )
         a = pi / afac
         umax = atan( a / sqrt(1. - a**2)  )
      end if

c Discard any unnecessary terrain points.  Test on the rate of change of slope,
c i.e., second derivative.  If that is below 1.e-3 then discard that point.

      if( tr.itp .gt. 0 ) fter = .true.
      if( fter ) then

c Check that all terrain range points are increasing.

         do i = 1, tr.itp-1
            ip1 = i + 1
            if( tr.terx(ip1) .lt. tr.terx(i) ) then
               ierror = -17
               return
            end if
         end do

c Test to see that first range value is 0.

         if( tr.terx(1) .gt. 0. ) then
            ierror = -18
            return
         end if

c Test to see if the last range point in the profile meets or exceeds RMAX.  If
```

```
c not then return error code.

        if( tr.terx(tr.itp) .lt. vnp.rmax ) then
           if( ef.lerr6 ) then
              ierror = -6
              return
           else
              tr.itp = min(tr.itp + 1, mxter)
              tr.terx(tr.itp) = vnp.rmax*1.01
              tr.tery(tr.itp) = tr.tery(tr.itp - 1)
           end if
        end if

c Test to see if the unit number for the scratch file is already attached to
c another file.  If so, search for a unit number that is unattached.

        inquire( iscr, OPENED = lopen )
        do while (lopen)
           iscr = iscr + 1
           inquire( iscr, OPENED = lopen )
        end do

        open( iscr, status = 'SCRATCH')
        write(iscr,*) tr.terx(1), tr.tery(1)         ! Keep first point of
        do i = 2, tr.itp-1                           ! terrain profile.
           im1 = i - 1
           ip1 = i + 1
           xm1 = tr.terx(im1)
           ym1 = tr.tery(im1)
           xi = tr.terx(i)
           yi = tr.tery(i)
           xp1 = tr.terx(ip1)
           yp1 = tr.tery(ip1)

           dx1 = amax1( 1.e-3, xi - xm1 )
           dx2 = amax1( 1.e-3, xp1 - xi )

           sl1 = (yi - ym1) / dx1
           sl2 = (yp1 - yi) / dx2

           scd = sl2 - sl1          ! dx is taken to be 1 m

c If second derivative is large enough then keep this point.

           if( abs(scd) .GT. 1.e-3 ) write(iscr,*) xi, yi

        end do

        write(iscr,*) tr.terx(tr.itp), tr.tery(tr.itp)     !Keep last point
        rewind( iscr )                                     !in profile.

c Now the scratch file contains all the necessary points for the terrain
c profile. Go back and read them in the arrays TR.TERX(), TR.TERY().

        bugfix = 0.
        lflag = eof(iscr)
        tr.itp = 0
        do while( .not. lflag )
           tr.itp = tr.itp + 1
           read(iscr,*) tr.terx(tr.itp), tr.tery(tr.itp)
           if( tr.terx(tr.itp) .ge. vnp.rmax ) exit
           bugfix = 0.
           lflag = eof(iscr)
        end do

        close(iscr)
```

```
c Determine minimum height of terrain profile.  Then adjust entire terrain
c profile by this minimum height HMINTER such that this is the new 0 reference.

         hminter = vnp.hmax
         do i = 1, tr.itp
            yi = tr.tery(i)
            if( yi .lt. hminter ) hminter = yi
         end do

c Get maximum height of terrain, then adjust terrain elevations by height
c offset.

         htermax = 0.
         do i = 1, tr.itp
            if( tr.tery(i) .gt. htermax ) htermax = tr.tery(i)
            tr.tery(i) = tr.tery(i) - hminter
         end do

c Return error code if VNP.HMAX does not exceed the maximum height of the
c terrain profile.

         if( htermax .gt. vnp.hmax ) then
            ierror = -8
            return
         end if

         antref = sv.antht + tr.tery(1)

         do i = 1, tr.itp-1

            ip1 = i + 1
            y1 = tr.tery(i)
            x1 = tr.terx(i)
            y2 = tr.tery(ip1)
            x2 = tr.terx(ip1)

            xdif = x2 - x1
            ydif = y2 - y1
            xdif = amax1( xdif, 1.e-5 )
            slope = ydif / xdif

            slp(i) = slope

c Calculate angle made from each terrain point height to antenna height above
c reference (HMINTER).  Determine maximum propagation angle so that direct ray
c angle will clear highest peak.

            if( y1 .gt. antref ) then
               angle = atan( (y1-antref) / x1 )
               if( angle .gt. angu ) angu = angle
            end if

         end do

c  Add 1/2 degree to the angle that clears the highest peak.

         angu = angu + hdeg

      end if

      hmref = vnp.hmin - hminter
      htlim = vnp.hmax-hminter
      zlim = amax1( htlim, antref )

c Initialize refractivity arrays.

      call refinit( ef.lerr12, vnp.rmax, rf, IERROR )
```

```
      if( ierror .ne. 0 ) return

c Calculate gradients and other variables for use in ray tracing.

      do i = 1, lvlep-1
         rm1 = refdum(i)
         rm2 = refdum(i+1)
         h1 = htdum(i)
         h2 = htdum(i+1)
         grad = ( rm2 - rm1 ) / ( h2 - h1 )
         if( abs( grad ) .lt. 1.e-3 ) grad = sign( 1., grad )*1.e-3
         dmdh(i) = grad * 1.e-6        ! for ray trace formulas
      end do

      jls = 0
      rmatht = 0.
      do i = 1, lvlep-1
         if((antref .lt. htdum(i+1)).and.(antref .ge. htdum(i))) then
            jls = i
            rmatht = refdum(i) + (antref - htdum(i)) * dmdh(i)*1.e6
            exit
         end if
      end do

      rlim = .9 * vnp.rmax
      prang = vnp.propang * radc

c Calculate the critical angle and perform ray trace to get the maximum
c propagation angle such that you get coverage at all heights and ranges
c >= 90% of maximum range.  This is done for automatic angle calculation.

c Get minimum M-unit value of profile for all heights above transmitter height.

      j = jls + 1
      rmina = refdum(j)
      do i = j, lvlep
         if( refdum(i) .lt. rmina ) rmina = refdum(i)
      end do

c Get minimum M-unit value of profile for all heights below transmitter heights.

      rminb = refdum(jls)
      do i = jls, 1, -1
         if( refdum(i) .lt. rminb ) rminb = refdum(i)
      end do

c Get critical angle if the transmitter is within or above a duct.

      acrit = 0.
      acrit1 = 0.
      acrit2 = 0.
      delm1 = rmatht - rmina
      delm2 = rmatht - rminb
      if( delm1 .gt. 0. ) acrit1 = sqrt( 2.e-6 * delm1 )
      if( delm2 .gt. 0. ) acrit2 = sqrt( 2.e-6 * delm2 )
      acrit = amax1( acrit1, acrit2 ) + 1.e-4

      thetamax = acrit

      at = atan( (zlim-antref) / vnp.rmax )
      thetamax = amax1( angu, at, acrit )

c If user inputs non-zero propagation angle, use that value.

      if( prang .ge. 1.e-6 ) thetamax = prang

c Get THETAMAX based on shallowest reflected ray traced to reach maximum height
```

```
c and still be within 90% of maximum range (for smooth surface). For terrain case
c the direct ray angle is used.

      call tracea( tr, prang, acrit )

c Add buffer for filter region.

      thetamax = thetamax / .75

c Put lower limit on THETAMAX depending on frequency ( in MHz ):
c for 5000 <= SV.FREQ <= 9000,  THETAMAX >= .5 deg
c for 4100 <= SV.FREQ < 5000,  THETAMAX >= .6 deg
c for 2900 <= SV.FREQ < 4100,  THETAMAX >= .7 deg
c for 2500 <= SV.FREQ < 2900,  THETAMAX >= .8 deg
c for 1500 <= SV.FREQ < 2500,  THETAMAX >= .9 deg
c for 600 < SV.FREQ < 1500,  THETAMAX >= 1 deg
c for 400 < SV.FREQ <= 600,  THETAMAX >= 2 deg
c for 200 < SV.FREQ <= 400,  THETAMAX >= 3 deg
c for SV.FREQ <= 200,  THETAMAX >= 4 deg

      if( sv.freq .le. 9000. ) thetamax = amax1(thetamax, 8.72665e-3)
      if( sv.freq .lt. 5000. ) thetamax = amax1(thetamax, 1.047197e-2)
      if( sv.freq .lt. 4100. ) thetamax = amax1(thetamax, 1.22173e-2)
      if( sv.freq .lt. 2900. ) thetamax = amax1(thetamax, 1.396263e-2)
      if( sv.freq .lt. 2500. ) thetamax = amax1(thetamax, 1.570796e-2)
      if( sv.freq .lt. 1500. ) thetamax = amax1(thetamax, 1.745329e-2)
      if( sv.freq .le. 600. ) thetamax = amax1(thetamax, 3.4906585e-2)
      if( sv.freq .le. 400. ) thetamax = amax1(thetamax, 5.2359877e-2)
      if( sv.freq .le. 200. ) thetamax = amax1(thetamax, 6.981317e-2)

      if(( sv.polar .eq. 'V' ) .and. ( prang .le. 1.e-6 ))
     +  thetamax = 2. * thetamax

c Get FFT size based on THETAMAX

      call getfftsz( ZLIM )

c Maximize THETAMAX within determined FFT size for terrain cases and if
c using automatic angle calculation.

      if(( fter ) .and. (prang .le. 1.e-6)) then

c Use 74% of ZMAX instead of 75% to leave some slop and ensure the FFT size is
c not surpassed.

         if( .74*zmax .gt. zlim ) then
            thetafrac = thetalaunch / thetamax
            zmax = zlim / .74
            sthetamax = float(n) * wl * .5 / zmax

c Put upper limits on THETAMAX depending on frequency.

            if( sv.freq .gt. 1000. ) then
               sthetamax = amin1( sthetamax, sdeg10 )
            else
               sthetamax = amin1( sthetamax, sdeg15 )
            end if
            delz= wl * .5 / sthetamax
            thetamax = asin( sthetamax )
            zmax = float(n) * delz
            thetalaunch = thetafrac * thetamax
         end if
      end if

c Determine horizon range based on transmitter height and 0 receiver height
c by RHOR = 3572. * sqrt( 1.3333 * antref )
```

```
      rhor = 4124.5387 * sqrt( sv.antht )
      dr = 2. * fko * delz**2
      rkm = vnp.rmax * 1.e-3

c Determine range step.

      if( fter ) then
         dr = amin1( dr, 700. )
         if( rkm .ge. 5. ) rllim = 75.
         if( rkm .ge. 10. ) rllim = 90.
         if( rkm .ge. 15. ) rllim = 100.
         if( rkm .ge. 20. ) rllim = 110.
         if( rkm .ge. 30. ) rllim = 175.
         if( rkm .ge. 50. ) rllim = 200.
         if( rkm .ge. 75. ) rllim = 250.
         if( rkm .ge. 100. ) rllim = 300.
         dr = amax1( dr, rllim )
      else
         dr = amin1( dr, 1000. )
         dr = amax1( dr, 30. )
         if( vnp.rmax .ge. rhor ) dr = amax1( 300., dr )
      end if
      dr2 = .5 * dr

c Path loss constant.

      plcnst=20.*alog10(2.*fko)

c Initialize variables for free-space propagator phase calculations.

      delp = pi/zmax
      FNorm = 2. / N
      cnst = delp / fko
      nm1 = n - 1
      dz2 = 2. * delz

c Initialize variables and set-up filter array.

      no4 = n/4
      n34 = 3.* no4
      cn75 = 4.* pi / N
      do i = 0, no4
         fj= cn75 * float(i)
         filt(i) = .5 + .5 * cos(fj)
      end do

c Initialize dielectric ground constants for vertical polarization.

      ig = 1
      if( tr.igr .eq. 0 ) then
         tr.igr = 1
         tr.rgrnd(1) = 0.
         tr.igrnd(1) = 0
      end if

      if( sv.polar .eq. 'V' ) call dieinit( sv, tr )

c Initialize starter field.

      call xyinit( sv, tr )

c Transform to z-space.

      call fft( u )

c Initialize C1 and C2 for start of PE calculations - only for vertical
c polarization.  NOTE: THIS IS ONLY FOR SMOOTH SURFACE.
```

```
      if( sv.polar .eq. 'V' ) then
         c1c = cmplx( 0., 0. )
         c2c = cmplx( 0., 0. )
         do i = 0, n
            nmi = n - i
            ui = u(i)
            unmi = u(nmi)
            if(( i .eq. 0 ) .or. (i .eq. n )) then
               ui = .5 * ui
               unmi = .5 * unmi
            end if

            iv = mod( i, 2 )
            cx = rav(i)
            if( iv .eq. 1 ) cx = -rav(i)
            c1c = ui * rav(i)
            c2c = unmi * cx

            c1 = c1 + c1c
            c2 = c2 + c2c
         end do
         c1 = c1 * rk
         c2 = c2 * rk

      end if

      ylast = 0.
      if( fter ) ylast = tr.tery(1)

      ycurm = 0.
      rout = 0.
      ycur = 0.

c Define mesh array in height

      do i=0,n
         ht(i)= float(i)*delz
      end do

c If smooth surface, trace THETALAUNCH ray and store all heights at each
c output range step in array HLIM().

      call traceh( vnp.nrout )

c Determine the free-space propagator (p-space) arrays.

      call phase1

c If smooth surface and range-independent case then initialize all refractivity
c and z-space propagator arrays now.

      if( rf.nprof .eq. 1 ) call profref( hminter, 0 )
      if(( .not. fter ) .and. (rf.nprof .eq. 1 )) then
         call intprof
         call phase2
      end if

      end
```

8.1.1 Subroutine ANTPAT

```
c ************************** SUBROUTINE ANTPAT ******************************

c Module Name: ANTPAT

c Module Security Classification: UNCLASSIFIED

c Purpose: Determines the antenna pattern factor for angle passed to routine.

c Version Number: 1.5

c INPUTS:
c    Argument List: IPTRN, SANG
c    Common: AFAC, BW, ELV, PELEV, SBW, UMAX

c OUTPUTS:
c    Argument List: PATFAC

c Files Included: NONE

c Calling Routines: XYINIT

c Routines called:  NONE

c GLOSSARY: For common variables refer to main glossary.

c    Input Variables:
c      IPTRN = Type of antenna pattern.
c      SANG = Sine of angle for which antenna pattern is sought.

c    Output Variables:
c      PATFAC = Antenna pattern factor.

c    Local Variables:
c      ARG = Angle argument used for SINX/X and generic height-finder antenna
c            pattern
c      DIRANG = Sine of direct ray angle = abs(SANG)
c      PR = Sine of angle U relative to sine of elevation angle (i.e.,
c           sine(u) - sine(elv)
c      U = Angle for which antenna pattern is sought.
c      UDIF = Angle U relative to the antenna elevation angle (i.e., U-ELV)

      subroutine antpat( iptrn, sang, PATFAC )

      common / pattern / pelev, afac, bw, elv, umax, sbw

c In the following pattern definitions, "u" refers to the angle for which
c the antenna pattern is sought, and "u0" refers to the elevation angle.

c  IPTRN = 0 gives Omnidirectional antenna pattern factor : f(u) = 1

      patfac = 1.

      if( iptrn .gt. 1 ) then
         u = asin( sang )
         udif = u - elv
      end if

c  IPTRN = 1 gives Gaussian antenna pattern based on
c  f(p-p0) = exp(-w**2 * ( p-p0 )**2 ) / 4, where p = sin(u) and
c  p0 = sin(u0)

      if( iptrn .eq. 1 ) then
         pr = sang - pelev
```

```
      patfac = exp(-pr * pr * afac)

c IPTRN = 2 gives sin(x)/x pattern based on
c f(u-u0) = sin(x) / x where x = afac * sin(u-u0) for |u-u0| <= umax
c f(u-u0) = .03 for |u-u0| > umax
c IPTRN = 4 gives height-finder pattern which is a special case of sin(x)/x

      elseif(( iptrn .eq. 2 ) .or. ( iptrn .eq. 4 )) then
         if( iptrn .eq. 4 ) then
            dirang = abs( sang )
            if( dirang .gt. elv ) udif = u - dirang
         end if
         if( abs(udif) .le. 1.e-6 ) then
            patfac = 1.
         elseif( abs( udif ) .gt. umax ) then
            patfac = .03
         else
            arg = afac * sin( udif )
            patfac = amin1( 1., amax1( .03, sin( arg ) / arg ) )
         end if

c IPTRN = 3 gives csc-sq pattern based on
c f(u) = 1 for u-u0 <= bw
c f(u) = sin(bw) / sin(u-u0) for u-u0 > bw
c f(u) = maximum of .03 or [1+(u-u0)/bw] for u-u0 < 0

      elseif( iptrn .eq. 3 ) then
         if( udif .gt. bw ) then
            patfac = sbw / sin( udif )
         elseif( udif .lt. 0 ) then
            patfac = amin1( 1., amax1( .03, (1. + udif/bw) ) )
         end if
      end if

      end
```

8.1.2 Subroutine REFINIT

```
c ************************* SUBROUTINE REFINIT *****************************

c Module Name: REFINIT

c Module Security Classification: UNCLASSIFIED

c Purpose: Initializes refractivity arrays used for subsequent PE
c          calculations.

c Version Number: 1.5

c INPUTS:
c   Argument List: ELERR12, RF structure, VRMAX,
c   Common: NONE

c OUTPUTS:
c   Argument List: IERROR
c   Common: HTDUM(), IS, LVLEP, REFDUM(), RV2

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT

c Routines called: REMDUP

c GLOSSARY: For common variables refer to main glossary

c   Input Variables:
c     ELERR12 = Element of user-provided error flag structure EF that will
c               trap on certain errors if set to .TRUE.  Refer to user's
c               manual.
c     RF = Refractivity structure for external environmental data elements.
c       RF.LVLEP = Number of levels in refractivity profile (for range
c                  dependent case all profiles must have same number of
c                  levels).
c       RF.REFMSL(,) = 2-dimensional array containing refractivity with
c                      respect to mean sea level of each profile.  Array
c                      format must be REFMSL(I,J) = M-unit value at Ith
c                      level of Jth profile.  J = 1 for range-independent
c                      cases.
c       RF.HMSL(,) = 2-dimensional array containing heights in meters with
c                    respect to mean sea level of each profile.  Array format
c                    must be HMSL(I,J) = height of Ith level of Jth profile.
c                    J = 1 for range-independent cases.
c       RF.RNGPROF() = Ranges of each profile in meters, i.e., RNGPROF(I) =
c                    range of Ith profile.  RNGPROF(1) should always be
c                    equal to 0.
c       RF.NPROF = Number of profiles.  Equals 1 for range-independent cases.
c     VRMAX = Maximum range in meters.

c   Output Variables:
c     IERROR = -12 -> Range of last refractivity profile entered (for range
c              dependent case) is less than VRMAX. (This is returned from
c              subroutine REFINIT).  Will only return this error if error flag
c              ELERR12 is set to .TRUE.).

c   Local Variables:
c     GRAD = Gradient of current refractivity level.
c     HDIF = Height difference between last two differing height levels in
c            each refractivity profile.
c     HLARGE = This is the maximum height at which the refractivity profile
c              is extrapolated.
c     LVLM1 = Last user-specified level in refractivity profile
```

```
c     LVLM2 = Second-to-last user-specified level in refractivity profile,
c             (i.e, LVLM1-1).
c     NP = Final number of refractivity profiles
c     RLARGE = This is the maximum range at which the refractivity profile
c              is "extrapolated".  For range-dependent case, the last entered
c              profile is then forced to be homogeneous to a range of RLARGE.

      subroutine refinit( elerr12, vrmax, rf, IERROR )

      include 'tpem.inc'

      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
     +                   profint(0:maxpts), ht(0:maxpts), is, lvlep

      record / refractivity / rf

      logical elerr12

      data hlarge/ 1.e6 /
      data rlarge / 1.e10 /

      ierror = 0

c Test to see if last profile entered ( for range dependent case ) meets or
c exceeds VRMAX, otherwise, return error (unless error trapping is turned off
c - EF.LERR12 = .FALSE.).

      if( rf.nprof .gt. 1 ) then
         if(( rf.rngprof(rf.nprof) .lt. vrmax ) .and. ( elerr12 )) then
            ierror = -12
            return
         end if
      end if

c Add extra level to tabulated profiles with extrapolated gradient.  Test on
c HDIF greater than 0 for profiles that contain multiple height/M-unit values
c that are equal.

      rf.lvlep = rf.lvlep + 1
      do i = 1,rf.nprof
         hdif = 0.
         lvlm1 = rf.lvlep
         lvlm2 = rf.lvlep
         do while( hdif .le. 1.e-6 )
            lvlm1 = lvlm1 - 1
            lvlm2 = lvlm1 - 1
            hdif = rf.hmsl(lvlm1,i) - rf.hmsl(lvlm2,i)
         end do
         grad = (rf.refmsl(lvlm1,i)-rf.refmsl(lvlm2,i)) / hdif

c If last gradient in refractivity profile is negative then return error.

         if( grad .lt. 0 ) then
            ierror = -14
            return
         end if

         rf.hmsl(rf.lvlep, i) = hlarge
         rf.refmsl( rf.lvlep, i ) = (hlarge-rf.hmsl(lvlm1,i)) * grad +
     +                   rf.refmsl( lvlm1, i )
      end do

      is = 1
      rv2=rf.rngprof(is)

      do i = 1, rf.lvlep
         refdum(i) = rf.refmsl( i, is )
```

```
   htdum(i) = rf.hmsl( i, is )
end do

np = rf.nprof + 1
rf.rngprof(np) = rlarge
do i = 1, rf.lvlep
   npm1 = np - 1
   rf.hmsl( i, np ) = rf.hmsl( i, npm1 )
   rf.refmsl( i, np ) = rf.refmsl( i, npm1 )
end do

lvlep = rf.lvlep
call remdup
end
```

8.1.3 Subroutine TRACEA

```
c *************************** SUBROUTINE TRACEA ****************************

c Module Name: TRACEA

c Module Security Classification: UNCLASSIFIED

c Purpose:  This routine performs a ray trace to determine the minimum angle
c           required (based on the reflected ray) to obtain a PE solution for
c           all heights up to ZLIM and all ranges beyond RLIM. THETAMAX
c           is then determined from this angle.  This is done only for smooth
c           surface and automatic angle calculation.  For terrian cases,
c           THETAMAX has already been set to the larger of the critical angle
c           (if a duct exists), the angle that clears the highest terrain
c           peak, and the tangent angle determined from HMAX and RMAX.

c           If PRANG is not equal to 0, then the user has overriden the
c           default calculation and THETAMAX is then determined based on
c           PRANG.  However a ray trace must still be done in order to
c           determine the initial launch angle such that the local angle of
c           the ray remains less than PRANG. The initial launch angle is
c           used in subroutine TRACEH.

c Version Number: 1.5

c INPUTS:
c   Argument List: ACRIT, PRANG, TR structure
c   Common: ANTREF, DMDH(), FTER, HTDUM(), JLS, LVLEP, RLIM,
c           SLP(), THETAMAX, ZLIM

c OUTPUTS:
c   Argument List: NONE
c   Common: RPE, THETALAUNCH, THETAMAX

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT

c Routines called: NONE

c GLOSSARY: For common variables refer to main glossary

c   Input Variables:
c     ACRIT = Critical angle, i.e., angle above which no rays are trapped
c             for ducting environment.
c     PRANG = User-specified propagation angle in radians.
c     TR = Terrain structure for external terrain data elements.
c       TR.TERX() = range points of terrain profile in meters.
c       TR.TERY() = height points of terrain profile in meters.

c   Output Variables:
c       For common variables refer to main glossary

c   Local Variables:
c     A0 = Angle in radians of ray before trace step.
c     A1 = Angle in radians of ray after trace step.
c     AMXCUR = Maximum of local angle along ray.
c     AMXCURL = Last (or previous) AMXCUR.
c     AS = Starting launch angle in radians.
c     ASL = Last (or previous) starting launch angle.
c     DEG15 = 15 degrees (in radians) - used as a maximum limit for
c             THETAMAX and THETALAUNCH.
c     GRAD = Gradient of current refractivity layer.
c     H0 = Height of ray in meters of ray before trace step.
```

```
c     H1 = Height of ray in meters of ray after trace step.
c     IDN = Integer with value of + or - 1.  Used to increment or decrement
c           initial launch angle.
c     ISET = Flag to test whether or not to stop main loop.
c     JL = Index of current refractivity layer ray tracing through.
c     LOOP = Flag to test whether or not to stop nested loop of ray tracing
c            individual ray.
c     RAD = Radicand for ray trace formula
c     R0 = Range of ray in meters before trace step.
c     R1 = Range of ray in meters after trace step.
c     RREF = Range at which traced ray is reflected.

      subroutine tracea( tr, prang, acrit )

      include 'tpem.inc'

      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
      common / trvar / dmdh(mxlvls), zlim, jls, thetalaunch, rlim
      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
     +                   profint(0:maxpts), ht(0:maxpts), is, lvlep

      record / terrain / tr

      complex qi

      logical fter, loop

      data deg15 / .2617994 /   !15 degrees in radians

c All heights and ranges are in meters, gradients are in M-unit/meter * 1.e-6
c and angles are in radians.

c Define in line ray trace functions:

      rada1( a, b ) = a**2 + 2. * grad * b              !a=a0, b=h1-h0
      rp( a, b ) = a + b / grad                         !a=r0, b=a1-a0
      ap( a, b ) = a + b * grad                         !a=a0, b=r1-r0
      hp( a, b, c ) = a + ( b**2 - c**2 ) / 2. / grad   !a=h0, b=a1, c=a0

      as = -thetamax
      idn = -1
      if( fter ) then
         as = thetamax
         if( prang .le. 1.e-6 ) idn = 1

c For initial shallow slope and non-zero user-defined maximum propagation
c angle, determine the range at which ray is reflected (RREF).  If this is
c less than the range of the 2nd terrain point, then treat as if this were a
c smooth surface problem.  I.e., determine THETAMAX based on reflected ray,
c not direct ray.

         if(( prang .gt. 1.e-6 ) .and. ( slp(1) .le. 1.e-6 )) then
            rref = (antref - tr.tery(1)) / tan( thetamax )
            if( rref .lt. tr.terx(2) ) then
               idn = -1
               as = -thetamax
            end if
         end if
      end if
      asl = 0.
      amxcurl = 0.
      iset = 0

      do while( iset .eq. 0 )
```

```
c Decrease or increase angle by 1 mrad, depending on value of IDN.
c Initialize ray trace variables.

         as = as + idn*1.e-3
         h0 = antref
         r0 = 0.
         rpe = 0.
         a0 = as
         jl = jls
         amxcur = 0.
         loop = .true.

c Perform ray trace until ray has reached ZLIM and/or RLIM where
c ZLIM = maximum of HMAX-HMINTER or ANTREF.
c RLIM = .9 * RMAX

         do while( loop )

            grad = dmdh(jl)
            if( a0 .lt. 0. ) h1 = htdum(jl)
            if( a0 .gt. 0. ) h1 = htdum(jl + 1)
            if( a0 .eq. 0. ) then
               if( grad .lt. 0. ) h1 = htdum(jl)
               if( grad .gt. 0. ) h1 = htdum(jl+1)
            end if
            if( h1 .gt. zlim ) h1 = zlim
            rad = radal( a0, h1-h0 )
            if( rad .gt. 0 ) then
               a1 = sign( 1., a0 ) * sqrt( rad )
            else
               a1 = 0.
               h1 = hp( h0, a1, a0 )
            end if

            r1 = rp( r0, a1-a0 )

            if(( a1 .le. 0. ) .and. ( h1 .le. htdum(jl) )) then
               h1 = htdum(jl)
               rad = radal( a0, h1-h0 )
               a1 = -sqrt( rad )
               r1 = rp( r0, a1-a0 )
               jl = jl - 1
               if( jl .eq. 0 ) jl = 1
            elseif(( a1 .ge. 0. ).and.( h1 .ge. htdum(jl+1) ))then
               h1 = htdum(jl+1)
               rad = radal( a0, h1-h0 )
               a1 = sqrt( rad )
               r1 = rp( r0, a1-a0 )
               jl = jl + 1
               if( jl .gt. lvlep ) jl = lvlep
            end if

            if( h1 .gt. zlim ) then
               h1 = zlim
               rad = radal( a0, h1-h0 )
               a1 = sqrt( rad )
               r1 = rp( r0, a1-a0 )
            end if

            h0 = h1
            r0 = r1
            a0 = a1

c Set RPE to range at which reflected ray hits the ground.

            if( h0 .le. 1.e-4 ) then
               a0 = -a0
```

```
            rpe = r0
         end if

c If A0 greater than 90 degrees then exit loop.

         if( a0 .ge. 1.57079 ) exit
         amxcur = amax1( amxcur, a0 )
         if(( h0 .ge. zlim ) .and. ( a0 .gt. 0.)) loop = .false.
         if( r0 .gt. rlim ) loop = .false.

      end do

c Test to see if the current ray traced from launch angle AS meets criteria.
c If ray traced does not reach ZLIM AND is not within RLIM then the initial
c launch angle AS is increased by 1 mrad and ray trace is
c repeated.  This is done only for smooth surface.

      if(( r0 .le. rlim ) .and. ( rpe .gt. 0. )) iset = 1

c If criteria is met then (if user specified a problem angle) make sure the
c local maximum angle is just within PRANG.  If not then repeat ray trace
c until this occurs.

      if( fter )then
         if( prang .gt. 1.e-6 ) then
            iset = 1
            if( amxcur .gt. thetamax ) iset = 0
            if( as .le. acrit+1.e-3 ) iset = 1   !Don't let launch angle
                                                 !be less than critical
                                                 !angle.
         else
            if(( r0 .le. rlim ) .and. ( h0 .ge. zlim )) iset = 1
            if( iset .eq. 1 ) thetamax = amax1( abs(as), amxcur )
         end if
      else
         if(( prang .gt. 1.e-6 ) .and. (iset .eq. 1)) then
            a = amax1( abs(as), amxcur )
            if( a .lt. prang ) then
               iset = 0
            elseif( asl .ne. 0. ) then
               as = asl
               amxcur = amxcurl
            end if
         end if
      end if

c Just as a safeguard - set absolute maximum of launch angle to 15 degrees.

      if( as .le. -deg15 ) then
         iset = 1
         as = -deg15
         amxcur = deg15
      end if

      asl = as
      amxcurl = amxcur

   end do

   if( .not. fter ) thetamax = amax1( abs(as), amxcur )
   thetalaunch = abs(as)

   end
```

8.1.4 Subroutine DIEINIT

```
c ************************* SUBROUTINE DIEINIT ***************************

c Module Name: DIEINIT

c Module Security Classification: UNCLASSIFIED

c Purpose: This routine calculates Conductivity and Permittivity
c          as a function of frequency in MHz.  All equations and coef-
c          ficients were obtained by using a SUMMASKETCH digitizer with
c          the CCIR volume 5 curves on page 74.  The digitized data was
c          then used with the TABLECURVE software to obtain the best fit
c          equations and coefficients used in this subroutine.  In some
c          cases two sets of equations were required to obtain a decent
c          fit across the 100 MHz - 100GHz range.  These curves fit the
c          digitized data to within 5%.

c Version Number: 1.5

c INPUTS:
c    Argument List: SV structure, TR structure
c    Common: NONE

c OUTPUTS: TR.DIELEC(,)

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT

c Routines called: NONE

c GLOSSARY:
c   Input Variables:
c      SV = System structure for external system data elements.
c         SV.FREQ = Frequency in MHz.
c      TR = Terrain structure for external terrain data elements.
c         TR.IGR = Number of different ground types specified.
c         TR.IGRND() = Type of ground composition for given terrain profile -
c                      can vary with range.  Different ground types are:
c                      0 = sea water, 1 = fresh water, 2 = wet ground,
c                      3 = medium dry ground, 4 = very dry ground,
c                      5 = user defined (in which case, values of relative
c                      permittivity and conductivity must be given).
c        TR.RGRND() = Ranges at which the ground types apply.

c   Output Variables:
c        TR.DIELEC(,) = 2-dimensional array containing the relative
c                       permittivity and conductivity; DIELEC(1,i) and
c                       DIELEC(2,i), respectively. Only needs to be specified
c                       if using IGRND(i) = 5, otherwise, TPEM will
c                       calculate based on frequency and ground types 0-4.

c   Local Variables:
c      EPSILON = Relative permittivity.
c      SIGMA = Conductivity.

      subroutine dieinit( sv, tr )

      include 'tpem.inc'

      record / terrain / tr
      record / systemvar / sv

      dimension a(14), b(14), c(14), d(14), e(14), f(14)
```

```
      data (a(i),i=1,14) / 1.4114535e-2, 3.8586749, 79.027635,
     +                     -0.65750351, 201.97103, 857.94335,
     +                     915.31026, 0.8756665, 5.5990969e-3,
     +                     215.87521, .17381269, 2.4625032e-2,
     +                     -4.9560275e-2, 2.2953743e-4 /
      data (b(i),i=1,14) / -5.2122497e-8, -2.1179295e-5, -2.2083308e-5,
     +                     5.5620223e-5, -2.5539582e-3, -8.9983662e-5,
     +                     -9.4530022e-6, 4.7236085e-5, 8.7798277e-5,
     +                     -7.6649237e-5, 1.2655183e-4, 1.8254018e-4,
     +                     2.9876572e-5, -8.1212741e-7 /
      data (c(i),i=1,14) / 5.8547829e-11, 9.1253873e-4, -3.5486605e-4,
     +                     6.6113198e-4, 1.2197967e-2, 5.5275278e-2,
     +                     -4.0348211e-3, 2.6051966e-8, 6.2451017e-8,
     +                     -2.6151055e-3, -1.6790756e-9, -2.664754e-8,
     +                     -3.0561848e-10, 1.8045461e-9 /
      data (d(i),i=1,14) / -7.6717423e-16, 6.5727504e-10, 2.7067836e-9,
     +                     3.0140816e-10, 3.7853169e-5, 8.8247139e-8,
     +                     4.892281e-8, -9.235936e-13, -7.1317207e-12,
     +                     1.2565999e-8, 1.1037608e-14, 7.6508732e-12,
     +                     1.1131828e-15, -1.960677e-12 /
      data (e(i),i=1,14) / 2.9856318e-21, 1.5309921e-8, 8.210184e-9,
     +                     1.4876952e-9, -1.728776e-6, 0.0,
     +                     7.4342897e-7, 1.4560078e-17, 4.2515914e-16,
     +                     1.9484482e-7, -2.9223433e-20, -7.4193268e-16,
     +                     0.0, 1.2569594e-15 /
      data (f(i),i=1,14) / 0., -1.9647664e-15, -1.0007669e-14, 0., 0.,
     +                     0., 0., -1.1129348e-22, -1.240806e-20, 0.,
     +                     0., 0., 0., -4.46811e-19 /

      f1 = sv.freq
      f2 = f1 * f1
      f3 = f1 * f2
      f4 = f1 * f3
      f5 = f1 * f4
      f6 = f1 * f5
      f7 = f1 * f6
      f8 = f1 * f7
      f9 = f1 * f8

      do i = 1, tr.igr

         select case ( tr.igrnd(i) )

c Permittivity and conductivity for salt water.

            case( 0 )
               epsilon = 70.
               sigma = 5.
               m = 1
               m1 = m + 1
               if( f1 .gt. 2253.5895 ) epsilon = 1. / ( a(m) +
     +            b(m)*f1 + c(m)*f2 + d(m)*f3 + e(m)*f4 )
               if( f1 .gt. 1106.207 ) then
                  sigma = a(m1) + c(m1)*f1 + e(m1)*f2
                  sigma = sigma / ( 1.+ b(m1)*f1 + d(m1)*f2 + f(m1)*f3 )
               end if

c Permittivity and conductivity for fresh water.

            case( 1 )
               epsilon = 80.0
               m = 3
               m1 = m + 1
               IF( f1 .gt. 6165.776 ) THEN
                  epsilon = a(m) + c(m)*f1 + e(m)*f2
                  epsilon = epsilon/(1. + b(m)*f1 + d(m)*f2 + f(m)*f3 )
```

```
                  end if
                  IF( f1 .gt. 5776.157) THEN
                     k = 2
                  else
                     m1 = m1 + 1
                     k = -1
                  end if
                  sigma = a(m1) + c(m1)*f1 + e(m1)*f2
                  sigma = (sigma / (1. + b(m1)*f1 + d(m1)*f2))**k

c  Permittivity and conductivity for wet ground.

               case( 2 )
                  epsilon = 30.0
                  m = 6
                  IF( f1 .ge. 4228.11 ) m = 7
                  if( f1 .gt. 1312.054 ) then
                     epsilon = a(m) + c(m)*f1 + e(m)*f2
                     epsilon = SQRT( epsilon / (1. + b(m)*f1 + d(m)*f2) )
                  end if
                  IF( f1 .gt. 15454.4) then
                     m1 = 8
                     g = 3.3253339e-28
                  else
                     m1 = 9
                     g = 1.3854354e-25
                  end if
                  sigma = a(m1) + b(m1)*f1 + c(m1)*f2 + d(m1)*f3 + e(m1)*f4
                  sigma = sigma + f(m1)*f5 + g*f6

c Permittivity and conductivity for medium dry ground.

               case( 3 )
                  epsilon = 15.0
                  IF( f1 .gt. 4841.945) THEN
                     m = 10
                     epsilon = a(m) + c(m)*f1 + e(m)*f2
                     epsilon = SQRT( epsilon / (1. + b(m)*f1 + d(m)*f2) )
                  end if
                  m1 = 12
                  IF( f1 .gt. 4946.751) m1 = 11
                  sigma = (a(m1) + b(m1)*f1 + c(m1)*f2 + d(m1)*f3 +
     +                  e(m1)*f4)**2

c Permittivity and conductivity for very dry ground.

               case( 4 )
                  epsilon = 3.0
                  IF( f1 .lt. 590.8924 ) then
                     sigma = 1.0e-4
                  else
                     IF( f1 .gt. 7131.933) THEN
                        m1 = 13
                        sigma = (a(m1) + b(m1)*f1 + c(m1)*f2 + d(m1)*f3)**2
                     else
                        m1 = 14
                        g = 9.4623158e-23
                        h = -1.1787443e-26
                        s = 7.9254217e-31
                        t = -2.2088286e-35
                        sigma = a(m1) + b(m1)*f1 + c(m1)*f2 + d(m1)*f3
                        sigma = sigma + e(m1)*f4 + f(m1)*f5 + g*f6
                        sigma = sigma + h*f7 + s*f8 + t*f9
                     end if
                  end if

c User-defined
```

```
            case( 5 )
               epsilon = tr.dielec(1,i)
               sigma = tr.dielec(2,i)

            case default
               ! Do nothing
         end select

         tr.dielec(1,i) = epsilon
         tr.dielec(2,i) = sigma
      end do

c Set dielectric constants equal to last provided ground constants at 1e7 km.

      tr.igr = tr.igr + 1
      tr.rgrnd(tr.igr) = 1.e10
      tr.dielec(1,tr.igr) = epsilon
      tr.dielec(2,tr.igr) = sigma

      end
```

8.1.5 Subroutine GETFFTSZ

```
c ************************** SUBROUTINE GETFFTSZ ***************************

c Module Name: GETFFTSZ

c Module Security Classification: UNCLASSIFIED

c Purpose:  Determines the FFT size needed for a given problem.

c Version Number: 1.5

c INPUTS:
c   Argument List: ZLIM
c   Common:  FTER, THETAMAX, WL
c   Parameter: MXNFFT

c OUTPUTS:
c   Argument List: ZLIM
c   Common: DELZ, LN, N, ZMAX

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT

c Routines called: NONE

c GLOSSARY: For common variables refer to main glossary.  For parameters
c           refer to TPEM.INC and FFTSIZ.INC

c   Input Variables:
c     ZLIM = Maximum height region where PE solution is valid = .75 * ZMAX.

c   Output Variables:
c     ZLIM = Calculates a new height limit equal to .75*ZMAX only if the
c            maximum transform size needed is too large to do specified
c            problem.  Fixes transform size to maximum and adjusts ZMAX and
c            ZLIM accordingly.

c   Local Variables:
c     STHETAMAX = Sine of THETAMAX

      subroutine getfftsz( ZLIM )

      include 'tpem.inc'

      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1

      logical fter

      complex qi

      sthetamax = sin( thetamax )
      delz= wl * .5 / sthetamax

c Set lower FFT limit to 2**9 for smooth surface cases, if terrain case then
c set lower FFT limit to 2**10.

      ln = 9
      if( fter ) ln=10
      N=2**LN
```

```
      zmax=delz*float(n)

c Determine transform size needed to perform calculations to a height of ZLIM,
c up to the maximum FFT size allowed.

      do while( .75*zmax .lt. zlim )
         ln = ln + 1
         if( ln .gt. mxnfft ) exit
         n = 2**ln
         zmax = delz * float(n)
      end do

c If the transform size needed is too large then set ZMAX and ZLIM
c accordingly.

      if( ln .gt. mxnfft ) then
         ln = mxnfft
         n = 2**ln
         zmax = delz * float(n)
         zlim = .75 * zmax
      end if

      end
```

8.1.6 Subroutine XYINIT

```
c ****************************** SUBROUTINE XYINIT ************************

c Module Name: XYINIT

c Module Security Classification: UNCLASSIFIED

c Purpose:  Determines the initial PE starter field.

c Version Number: 1.5

c INPUTS:
c   Argument List: SV structure, TR structure
c   Common: DELP, FILT(), FKO, N, N34, RNG2, WL, ZMAX

c OUTPUTS:
c   Argument List: NONE
c   Common: RNG2, U()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT

c Routines called: ANTPAT, GETALN

c GLOSSARY: For common variables see main glossary

c   Input Variables:
c     SV = System structure for external system data elements.
c       SV.ANTHT = transmitting antenna height above local ground in meters.
c       SV.IPAT = integer value indicating type of antenna pattern desired.
c                 IPAT = 0 -> omni
c                 IPAT = 1 -> gaussian
c                 IPAT = 2 -> sinc x
c                 IPAT = 3 -> csc**2 x
c                 IPAT = 4 -> generic height-finder
c       SV.POLAR = 1-character string indicating polarization.  H-horizontal,
c                  V-vertical.
c    TR = Terrain structure for external terrain data elements.

c  Output Variables:
c    For common variables refer to main glossary

c  Local Variables:
c    ANTKO = Exponential term in calculation of DTERM and RTERM
c    ATTN = Attenuation factor for filtering
c    CRAD = Square root term in reflection coefficient calculation
c    CTHETA = Cosine of angle PK
c    DTERM = Field due to real point source at height SV.ANTHT, i.e.,
c            DTERM=exp{-i*sv.antht*fko*sin(theta)}.
c    DTHETA = Bin width in angle- (or p-) space
c    FACD = Antenna pattern factor for direct ray angle.
c    FACR = Antenna pattern factor for reflected ray angle.
c    PK = Angle value for bin I, i.e., I*DTHETA
c    REFCOEF = Complex reflection coefficient.
c    RTERM = Field due to image point source at height -SV.ANTHT, i.e.,
c            RTERM=exp{i*sv.antht*fko*sin(theta)}.
c    SGAIN = Normalization factor.
c    SRAD = Sine term in reflection coefficient calculation
c    STHETA = Sine of angle PK
c    ZPK = Phase term for DTERM and RTERM

      SUBROUTINE xyinit( sv, tr )
```

```
      include 'tpem.inc'

      common / arrays / u(0:maxpts), filt(0:maxn4), frsp(0:maxpts),
     +                  envpr(0:maxpts), ulst(0:maxpts)
      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / impedance / alphav, rav(0:maxpts), rng, rng2, c1, c2,
     +                     rk, c1m, c2m, ig, root

      record / systemvar / sv
      record / terrain / tr

      logical fter

      complex u, frsp, envpr, ulst, qi, root
      complex alphav, rav, rng, rng2, c1, c2, rk, c1m, c2m
      complex refcoef, rterm, dterm, crad, srad

c Reflection coefficient is defaulted to -1 for horizontal polarization.

      refcoef = cmplx( -1., 0. )
      if( sv.polar .eq. 'V' ) call getaln( tr )

      sgain= sqrt( wl ) / zmax

      dtheta = delp / fko
      antko = fko * sv.antht

      DO I=0,N

         pk = float(i) * dtheta
         zpk = pk * antko

c Get antenna pattern factors for the direct and reflected rays.

         call antpat( sv.ipat, pk, FACD )
         call antpat( sv.ipat, -pk, FACR )

c If vertical polarization, then determine reflection coefficient.

         if( sv.polar .eq. 'V' ) then
            ctheta = cos( pk )
            stheta = sin( pk )
            crad = csqrt( rng2 - ctheta**2 )
            srad = rng2 * stheta
            refcoef = (srad - crad) / (srad + crad)
         end if

         rterm = cmplx( cos( zpk ), sin( zpk ) )
         dterm = conjg( rterm )

         u(i) = sgain * ( facd * dterm + refcoef * facr * rterm )

      end do

c Filter upper 1/4 of the field.

      do i = n34, n
         attn = filt(i-n34)
         u(i) = attn*u(i)
      end do

      END
```

8.1.7 Subroutine FFT

```
c ************************** SUBROUTINE FFT ********************************

c Module Name: FFT

c Module Security Classification: UNCLASSIFIED

c Purpose: Performs fast Fourier sine transform on complex PE field array U.

c Version Number: 1.5

c INPUTS:
c   Argument List: U()
c   Common: LN, N
c   Parameter: MAXPTS

c OUTPUTS:
c   Argument List: U()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT, PESTEP

c Routines Called: SINFFT

c GLOSSARY: For common variables refer to main glossary.  For parameters,
c           refer to TPEM.INC

c   Input Variables:
c     U() = Complex field to be transformed.

c   Output Variables:
c     U() = Transform of complex field.

c   Local Variables:
c     X() = Real part of field.
c     Y() = Imaginary part of field.

      subroutine fft( u )

      include 'tpem.inc'

      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1

      complex u(0:*)

      dimension x(0:maxpts), y(0:maxpts)

      do i = 0, n
         x(i) = real( u(i) )
         y(i) = imag( u(i) )
      end do

      call sinfft( ln, X )
      call sinfft( ln, Y )

      do i = 0, n
         u(i) = cmplx( x(i), y(i) )
      end do

      end
```

8.1.8 Subroutine SINFFT

```
SUBROUTINE SINFFT ( N, X )
C
C***********************************************************************
C*                                                                     *
C*                                                                     *
C* PURPOSE:   SINFFT replaces the real array X()                       *
C*            by its finite discrete sine transform                    *
C*                                                                     *
C* METHOD :                                                            *
C*                                                                     *
C*     The algorithm is based on a mixed radix (8-4-2) real vector     *
C*     fast Fourier synthesis routine published by Bergland:           *
C*                                                                     *
C*     ( G.D. Bergland, 'A Radix-eight Fast Fourier Transform          *
C*      Subroutine for Real-valued Series,' IEEE Transactions on       *
C*      Audio and Electro-acoustics', vol. AU-17, pp. 138-144, 1969 )  *
C*                                                                     *
C*     and sine and cosine transform algorithms for real series        *
C*     published by Cooley, Lewis, and Welch:                          *
C*                                                                     *
C*     (J.W. COOLEY, P.A.W. LEWIS AND P.D. WELSH, 'The Fast Fourier    *
C*     Transform Algorithm: Programming Considerations in the          *
C*     Calculation of Sine, Cosine and Laplace Transforms',            *
C*     J. SOUND VIB., vol. 12, pp. 315-337, 1970 ).                    *
C*                                                                     *
C*                                                                     *
C* ARGUMENTS:                                                          *
C*                       -- INPUT --                                   *
C*                                                                     *
C*      N...... transform size ( = 2**N )                              *
C*                                                                     *
C*      X().... data array dimensioned 2**N in calling program         *
C*                                                                     *
C*                      -- OUTPUT --                                   *
C*                                                                     *
C*      X().... sine transform                                         *
C*                                                                     *
C*  TABLES:  array     required size                                   *
C*                                                                     *
C*              B          2**N                                        *
C*              JINDX      2**(N-1)                                    *
C*              COSTBL     2**(N-4)                                    *
C*              SINTBL     2**(N-4)                                    *
C*                                                                     *
C*  Sub-programs called:  -                                            *
C*                                                                     *
C*                R8SYN..... (radix 8 synthesis)                       *
C*                                                                     *
C***********************************************************************
C
      include 'fftsiz.inc'

       INTEGER*4    N
C
       DIMENSION    X(0:*)
       INTEGER*4    NMAX2, NMAX16, NP, NPD2, NPD4
C
       PARAMETER    ( NMAX2  = MAXPTS/2 )
       PARAMETER    ( NMAX16 = MAXPTS/16 )
       DIMENSION    B(MAXPTS), JINDX (NMAX2)
       DIMENSION    SINES (MAXPTS)
       DIMENSION    COSTBL (NMAX16), SINTBL (NMAX16)
C
```

```
      SAVE B, COSTBL, JINDX, SINES, SINTBL
      SAVE NSAVE, N4, N8, NP, NPD2, NPD4, NPD16, NPM1
C
C
      DOUBLE PRECISION ARG, DT, PI
      DATA NSAVE  / 0 /
      DATA PI     / 3.1415 92653 58979 32D0 /
C
C----------------------------------------------------------------------+
C
      IF ( N .NE. NSAVE ) THEN
C                                  compute constants and construct tables
         NSAVE = N
         N8 = NSAVE / 3
         N4 = NSAVE - 3 * N8 - 1
         NP = 2**N
         NPD2  = NP / 2
         NPD4  = NP / 4
         NPD16 = NP / 16
         NPM1  = NP - 1
C                                       build reciprical sine table
         DT    = PI / FLOAT ( NP )
         DO 10 J = 1, NPM1
            ARG = DT * J
            SINES ( J ) = (0.5D0 / SIN ( ARG ))
   10    CONTINUE
C                                  construct bit reversed subscript table
         J1 = 0
         DO 30 J = 1, NPD2 - 1
            J2 = NPD2
   20       CONTINUE
            IF ( IAND ( J1, J2 ) .NE. 0 ) THEN
               J1 = IABS ( J1 - J2 )
               J2 = J2 / 2
               GO TO 20
            ENDIF
            J1 = J1 + J2
            JINDX ( J ) = J1
   30    CONTINUE
C
C                           form the trig tables for the radix-8 passes;
C                           tables are stored in bit reversed order.
         J1 = 0
         DO 50 J = 1, NPD16 - 1
            J2 = NPD16
   40       CONTINUE
            IF ( IAND ( J1, J2 ) .NE. 0 ) THEN
               J1 = IABS ( J1 - J2 )
               J2 = J2 / 2
               GO TO 40
            ENDIF
            J1  = J1 + J2
            ARG = DT * FLOAT (J1)
            COSTBL ( J ) =  COS ( ARG )
            SINTBL ( J ) = -SIN ( ARG )
   50    CONTINUE
C
      ENDIF
C
C                      ***  form the input Fourier coefficients ***
C
C                                                        sine transform
         B ( 1 ) = -2. * X ( 1 )
         B ( 2 ) =  2. * X ( NPM1 )
         J1 = 0
         DO 110 J = 3, NPM1, 2
```

```
            J1 = J1 + 1
            J2 = JINDX ( J1 )
            B ( J )     = X ( J2 - 1 ) - X ( J2 + 1 )
            B ( J + 1 ) = X ( NP-J2 )
  110    CONTINUE
C
C                 ***************************************
C                 *                                     *
C                 *    Begin Fast Fourier Synthesis     *
C                 *                                     *
C                 ***************************************
C
      IF ( N8 .NE. 0 ) THEN
C                                                 radix-8 iterations
         INTT = 1
         NT = NPD16
         DO 130 J = 1, N8
            J1 =  1 + INTT
            J2 = J1 + INTT
            J3 = J2 + INTT
            J4 = J3 + INTT
            J5 = J4 + INTT
            J6 = J5 + INTT
            J7 = J6 + INTT
C***
            CALL R8SYN (INTT, NT, COSTBL, SINTBL, B(1), B(J1), B(J2),
     *                      B(J3), B(J4), B(J5), B(J6), B(J7)  )
C***
            NT = NT / 8
            INTT = 8 * INTT
  130    CONTINUE
      ENDIF
C                                                 radix-4 iteration
      IF ( N4 .GT. 0 ) THEN
         J1 = NPD4
         J2 = 2*NPD4
         J3 = 3*NPD4
         DO 140 J = 1, NPD4
            T0 = B(J) + B(J + J1)
            T1 = B(J) - B(J + J1)
            T2 = 2. * B(J + J2)
            T3 = 2. * B(J + J3)
            B(J)      = T0 + T2
            B(J + J2) = T0 - T2
            B(J + J1) = T1 + T3
            B(J + J3) = T1 - T3
  140    CONTINUE
C
      ELSE IF ( N4 .EQ. 0 ) THEN
C                                                 radix-2 iteration
         K = NPD2
         DO 150 J = 1, NPD2
            K    = K + 1
            T    = B(J) + B (K)
            B(K) = B(J) - B (K)
            B(J) = T
  150    CONTINUE
      ENDIF
C
C                 ************************
C                 *                      *
C                 *    Form Transform    *
C                 *                      *
C                 ************************
C
C
C                                                 sine transform
```

```
         J1 = NP
         DO 160 J = 1, NPM1
            X(J) = .25*(( B(J+1) + B(J1)) * SINES(J) - B(J+1) + B(J1))
              J1 = J1 - 1
  160    CONTINUE

      RETURN
      END
C
C
      SUBROUTINE R8SYN ( INTT, NT, COSTBL, SINTBL, B0, B1, B2, B3,
     *                   B4, B5, B6, B7 )
C
C*********************************************************************
C
C  PURPOSE:    Radix-8 synthesis subroutine used by mixed radix driver.
C
C
C
C*********************************************************************
C
C
      DIMENSION   COSTBL(*), SINTBL(*)
      DIMENSION B0(*), B1(*), B2(*), B3(*), B4(*), B5(*), B6(*), B7(*)
C
C
C           ///     Local variables     ///
C
C
      DOUBLE PRECISION C1, C2, C3, C4, C5, C6, C7
      DOUBLE PRECISION S1, S2, S3, S4, S5, S6, S7
      DOUBLE PRECISION CPI4, CPI8, R2, SPI8
C
      SAVE  CPI4, CPI8,  R2, SPI8
C
      DATA  R2    / 1.41421 35623 7310D+0 /,
     *      CPI4  / 0.70710 67811 8655D+0 /,
     *      CPI8  / 0.92387 95325 1129D+0 /,
     *      SPI8  / 0.38268 34323 6509D+0 /
C
C
C-------------------------------------------------------------------+
C
      JT = 0
      JL = 2
      JR = 2
      JI = 3
      INT8 = 8 * INTT
C
      DO 60 K = 1, INTT
        T0 = B0(K) + B1(K)
        T1 = B0(K) - B1(K)
        T2 = B2(K) + B2(K)
        T3 = B3(K) + B3(K)
        T4 = B4(K) + B6(K)
        T5 = B4(K) - B6(K)
        T6 = B7(K) - B5(K)
        T7 = B7(K) + B5(K)
        T8 = R2 * (T7 - T5)
        T5 = R2 * (T7 + T5)
        TT0 = T0 + T2
        T2  = T0 - T2
        TT1 = T1 + T3
        T3  = T1 - T3
        T4  = T4 + T4
        T6  = T6 + T6
C
```

```
         B0(K)  = TT0 + T4
         B4(K)  = TT0 - T4
         B1(K)  = TT1 + T5
         B5(K)  = TT1 - T5
         B2(K)  = T2 + T6
         B6(K)  = T2 - T6
         B3(K)  = T3 + T8
         B7(K)  = T3 - T8
   60 CONTINUE
C
        IF ( NT .EQ. 0 )                      RETURN
C
        K0 = INT8 + 1
        KLAST = INT8 + INTT
C
        DO 70 K = K0, KLAST
           T1 = B0(K) + B6(K)
           T3 = B0(K) - B6(K)
           T2 = B7(K) - B1(K)
           T4 = B7(K) + B1(K)
           T5 = B2(K) + B4(K)
           T7 = B2(K) - B4(K)
           T6 = B5(K) - B3(K)
           T8 = B5(K) + B3(K)
C
           B0(K)  = (T1 + T5) + (T1 + T5)
           B4(K)  = (T2 + T6) + (T2 + T6)
           T5     = T1 - T5
           T6     = T2 - T6
           B2(K)  = R2 * (T6 + T5)
           B6(K)  = R2 * (T6 - T5)
           T1     = T3 * CPI8 + T4 * SPI8
           T2     = T4 * CPI8 - T3 * SPI8
           T3     = T8 * CPI8 - T7 * SPI8
           T4     =  - T7 * CPI8 - T8 * SPI8
           B1(K)  = (T1 + T3) + (T1 + T3)
           B5(K)  = (T2 + T4) + (T2 + T4)
           T3     = T1 - T3
           T4     = T2 - T4
           B3(K)  = R2 * (T4 + T3)
           B7(K)  = R2 * (T4 - T3)
   70  CONTINUE
C
       DO 90 JT = 1, NT-1
        C1 = COSTBL(JT)
        S1 = SINTBL(JT)
        C2 = C1 * C1 - S1 * S1
        S2 = C1 * S1 + C1 * S1
        C3 = C1 * C2 - S1 * S2
        S3 = C2 * S1 + S2 * C1
        C4 = C2 * C2 - S2 * S2
        S4 = C2 * S2 + C2 * S2
        C5 = C2 * C3 - S2 * S3
        S5 = C3 * S2 + S3 * C2
        C6 = C3 * C3 - S3 * S3
        S6 = C3 * S3 + C3 * S3
        C7 = C3 * C4 - S3 * S4
        S7 = C4 * S3 + S4 * C3
C
        K  = JI * INT8
        J0 = JR * INT8 + 1
        JLAST = J0 + INTT - 1
C
        DO 80 J = J0, JLAST
C
           K   = K + 1
           TR0 = B0(J) + B6(K)
```

```
          TR1 = B0(J) - B6(K)
          TI0 = B7(K) - B1(J)
          TI1 = B7(K) + B1(J)
          TR2 = B4(K) + B2(J)
          TI3 = B4(K) - B2(J)
          TI2 = B5(K) - B3(J)
          TR3 = B5(K) + B3(J)
          TR4 = B4(J) + B2(K)
          T0  = B4(J) - B2(K)
          TI4 = B3(K) - B5(J)
          T1  = B3(K) + B5(J)
          TR5 = CPI4 * (T1 + T0)
          TI5 = CPI4 * (T1 - T0)
          TR6 = B6(J) + B0(K)
          T0  = B6(J) - B0(K)
          TI6 = B1(K) - B7(J)
          T1  = B1(K) + B7(J)
          TR7  =  - CPI4 * (T0 - T1)
          TI7  =  - CPI4 * (T0 + T1)
          T0   = TR0 + TR2
          TR2  = TR0 - TR2
          T1   = TI0 + TI2
          TI2  = TI0 - TI2
          T2   = TR1 + TR3
          TR3  = TR1 - TR3
          T3   = TI1 + TI3
          TI3  = TI1 - TI3
          T5   = TI4 + TI6
          TTR6 = TI4 - TI6
          TI6  = TR6 - TR4
          T4   = TR4 + TR6
          T7   = TI5 + TI7
          TTR7 = TI5 - TI7
          TI7  = TR7 - TR5
          T6   = TR5 + TR7
C
          B0(J) = T0 + T4
          B0(K) = T1 + T5
          B4(J) = C4 * (T0 - T4)    - S4 * (T1 - T5)
          B4(K) = C4 * (T1 - T5)    + S4 * (T0 - T4)
C
          B1(J) = C1 * (T2 + T6)    - S1 * (T3 + T7)
          B1(K) = C1 * (T3 + T7)    + S1 * (T2 + T6)
          B5(J) = C5 * (T2 - T6)    - S5 * (T3 - T7)
          B5(K) = C5 * (T3 - T7)    + S5 * (T2 - T6)
C
          B2(J) = C2 * (TR2 + TTR6) - S2 * (TI2 + TI6)
          B2(K) = C2 * (TI2 + TI6)  + S2 * (TR2 + TTR6)
          B6(J) = C6 * (TR2 - TTR6) - S6 * (TI2 - TI6)
          B6(K) = C6 * (TI2 - TI6)  + S6 * (TR2 - TTR6)
C
          B3(J) = C3 * (TR3 + TTR7) - S3 * (TI3 + TI7)
          B3(K) = C3 * (TI3 + TI7)  + S3 * (TR3 + TTR7)
          B7(J) = C7 * (TR3 - TTR7) - S7 * (TI3 - TI7)
          B7(K) = C7 * (TI3 - TI7)  + S7 * (TR3 - TTR7)
C
   80  CONTINUE
C
       JR = JR + 2
       JI = JI - 2
       IF ( JI .GT. JL) GOTO 90
       JI = JR + JR - 1
       JL = JR
   90 CONTINUE
C
      RETURN
      END
```

8.1.9 Subroutine TRACEH

```
c **************************** SUBROUTINE TRACEH *****************************

c Module Name: TRACEH

c Module Security Classification: UNCLASSIFIED

c Purpose:  Computes ray trace for a single ray with launch angle -THETALAUNCH
c           for smooth surface. For terrain case, launch angle is THETALAUNCH.
c           Upon reflection the heights of this ray at each output range point
c           RO is then stored in HLIM() for subsequent output of loss values
c           in array MLOSS. This is done so that only loss values that fall
c           within the valid PE solution region are output or passed back in
c           MLOSS.

c Version Number: 1.5

c INPUTS:
c   Argument List: NVROUT
c   Common: ANTREF, DMDH(), DROUT, FTER, HTDUM(), HTLIM, JLS, LVLEP,
c           RLIM, THETALAUNCH, ZLIM

c OUTPUTS:
c   Argument List: NONE
c   Common: HLIM(), RPE

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT

c Routines called: NONE

c GLOSSARY:  For common variables refer to main glossary

c   Input Variables:
c     NVROUT = Number of output range points.

c   Output Variables:
c     For common variables refer to main glossary

c   Local Variables:
c     A0 = Angle in radians of ray before trace step.
c     A1 = Angle in radians of ray after trace step.
c     GRAD = Gradient of current refractivity layer.
c     H0 = Height of ray in meters of ray before trace step.
c     H1 = Height of ray in meters of ray after trace step.
c     IHU = Range index at which the traced ray has reached the maximum
c           calculation height.
c     JL = Index of current refractivity layer ray tracing through.
c     R0 = Range of ray in meters before trace step.
c     R1 = Range of ray in meters after trace step.
c     RO = Current output range at which to store height of trace ray in
c          HLIM().

      subroutine traceh( nvrout )

      include 'tpem.inc'

      common / trvar / dmdh(mxlvls), zlim, jls, thetalaunch, rlim
      common / rhstps / dr, drout, dzout, dr2, zout(mxzout)
      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
```

```
     +                    profint(0:maxpts), ht(0:maxpts), is, lvlep

      complex qi
      logical fter

c Define one-line ray trace functions:

      radal( a, b ) = a**2 + 2. * grad * b
      rp( a, b ) = a + b / grad
      ap( a, b ) = a + b * grad
      hp( a, b, c ) = a + ( b**2 - c**2 ) / 2. / grad

      a0 = -thetalaunch
      if( fter ) a0 = thetalaunch
      h0 = antref
      jl = jls
      ro = drout
      ihu = 0
      r0 = 0.
      rpe = 0.

c Ray is traced through NVROUT output range points.

      do i = 1, nvrout

c Trace until ray reaches output range point RO.

         do while( r0 .lt. ro )

            r1 = ro

            grad = dmdh(jl)
            a1 = ap( a0, r1-r0 )

            if( sign(1.,a0) .ne. sign(1.,a1) ) then
               a1 = 0.
               r1 = rp( r0, a1-a0 )
            end if
            h1 = hp( h0, a1, a0 )

            if(( a1 .le. 0. ) .and. ( h1 .le. htdum(jl) ))  then
               h1 = htdum(jl)
               rad = radal( a0, h1-h0 )
               a1 = -sqrt( rad )
               r1 = rp( r0, a1-a0 )
               jl = jl - 1
               if( jl .eq. 0 ) jl = 1
            elseif(( a1 .ge. 0. ) .and. ( h1 .ge. htdum(jl+1) )) then
               h1 = htdum(jl+1)
               rad = radal( a0, h1-h0 )
               a1 = sqrt( rad )
               r1 = rp( r0, a1-a0 )
               jl = jl + 1
               if( jl .gt. lvlep ) jl = lvlep
            end if

            if( r1 .gt. ro ) then
               r1 = ro
               a1 = ap( a0, r1-r0 )
               h1 = hp( h0, a1, a0 )
            end if

            h0 = h1
            r0 = r1
            a0 = a1

            if( h0 .le. 1.e-4 ) then
```

```
               a0 = -a0
               rpe = r0
            end if

c If ray has reached ZLIM (maximum output height region) then all heights for
c subsequent output range points will also be at ZLIM - so can exit loop.

            if( h0 .gt. zlim ) then
               ihu = i
               exit
            end if
         end do

         if( ihu .gt. 0 ) exit

         if( a0 .lt. 0. ) hlim(i) = 0.
         if( a0 .ge. 0. ) hlim(i) = h0

         ro = ro + drout

      end do

      if( ihu .gt. 0 ) then
         do i = ihu, nvrout
            hlim(i) = htlim
         end do
      end if

      end
```

8.1.10 Subroutine PHASE1

```
c **************************** SUBROUTINE PHASE1 ***************************

c Module Name: PHASE1

c Module Security Classification: UNCLASSIFIED

c Purpose: Initialize free-space propagator array FRSP() using wide-angle
c          propagator.

c Version Number: 1.5

c INPUTS:
c   Argument List: NONE
c   Common: CNST, DR, FILT(), FKO, FNORM, N, N34

c OUTPUTS:
c   Argument List: NONE
c   Common: FRSP()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT

c Routines called: NONE

c GLOSSARY:

c   Input Variables:
c     For common variables refer to main glossary

c   Output Variables:
c     For common variables refer to main glossary

c   Local Variables:
c     AK = Term used in ANG for each bin (i.e., I*DELP/FKO)
c     AKSQ = Square of AK
c     ANG = Exponent term:
c           ANG = -i*dr*k*[1-sqrt(1-(p/k)**2)] where k is the free-space
c           wavenumber, p is the transform variable (p=k*sin(theta)), and
c           i is the imaginary number (i=sqrt(-1)).
c     ATTN = Attenuation factor for filtering.
c     CAK = Square root term in ANG
c     DRFK = Term used in ANG (i.e., DR*FKO)

      SUBROUTINE PHASE1

      include 'tpem.inc'

      common / arrays / u(0:maxpts), filt(0:maxn4), frsp(0:maxpts),
     +                  envpr(0:maxpts), ulst(0:maxpts)
      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / rhstps / dr, drout, dzout, dr2, zout(mxzout)

      logical fter

      complex u, frsp, envpr, ulst, qi

      double precision cak

      drfk = dr * fko
```

```
      DO I=0,N
         ak = float(i) * cnst
         aksq=ak * ak
         aksq = amin1( 1., aksq )
         cak = sqrt(1. - aksq)
         ang = drfk * ( 1.d0 - cak )
         ca = cos( ang )
         sa = -sin( ang )
         frsp(i) = fnorm * cmplx( ca, sa )
      end do

c Filter the upper 1/4 of the propagator arrays.

      do  i = n34, n
         attn = filt(i-n34)
         frsp(i) = attn * frsp(i)
      end do

      END
```

8.1.11 Subroutine PHASE2

```
c ************************** SUBROUTINE PHASE2 *****************************

c Module Name: PHASE2

c Module Security Classification: UNCLASSIFIED

c Purpose: Calculates the environmental phase term for a given profile, then
c          stores in array ENVPR().

c Version Number: 1.5

c INPUTS:
c   Argument List: NONE
c   Common: DR, FILT(), N, N34, PROFINT()

c OUTPUTS:
c   Argument List: NONE
c   Common: ENVPR()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT, PESTEP

c Routines called: NONE

c GLOSSARY:

c   Input Variables:
c     For common variables refer to main glossary

c   Output Variables:
c     For common variables refer to main glossary

c   Local Variables:
c     ANG = Exponential argument in determining ENVPR() (i.e., DR*FKO*M*1.e-6)
c           where M is the M-unit value
c     ATTN = Attenuation factor for filtering

      SUBROUTINE PHASE2

      include 'tpem.inc'

      common / rhstps / dr, drout, dzout, dr2, zout(mxzout)
      common / arrays / u(0:maxpts), filt(0:maxn4), frsp(0:maxpts),
     +                  envpr(0:maxpts), ulst(0:maxpts)
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
     +                   profint(0:maxpts), ht(0:maxpts), is, lvlep

      complex u, frsp, envpr, ulst

      do i = 0, n
         ang = dr * profint(i)
         ca = cos( ang )
         sa = sin( ang )
         envpr(i) = cmplx( ca, sa )
      end do

c Filter upper 1/4 of the arrays.

      do i = n34, n
         attn = filt(i-n34)
         envpr(i) = attn * envpr(i)
```

```
end do
END
```

8.1.12 Subroutine PROFREF

```
c ****************************** SUBROUTINE PROFREF *************************

c Module Name: PROFREF

c Module Security Classification: UNCLASSIFIED

c Purpose: This subroutine determines the refractivity profile with respect
c          to the reference height YREF which, depending on the value of IFLAG,
c          can be HMINTER or the local ground height above HMINTER.

c Version Number: 1.5

c INPUTS:
c   Argument List: IFLAG, YREF
c   Common: HTDUM(), LVLEP, REFDUM()
c   Parameter: MXLVLS

c OUTPUTS:
c   Argument List: NONE
c   Common: HREF(), NLVL, REFREF()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines:  PEINIT, REFINTER

c Routines called: NONE

c GLOSSARY: For common variables refer to main glossary.  For parameters refer
c           to FFTSIZ.INC and TPEM.INC

c   Input Variables:
c     IFLAG = 0: Profile arrays REFREF() and HREF() will be referenced to
c                height HMINTER, and will also be used to initialize REFDUM()
c                and HTDUM().
c           = 1: Profile arrays REFREF() and HREF() will be referenced to the
c                local ground height.
c     YREF = Reference height in meters at current range.

c   Output Variables:
c     For common variables refer to main glossary

c   Local Variables:
c     FRAC = Fractional height over which to interpolate
c     IBMSL = Flag indicating if YREF is below mean sea level (msl)
c             IBMSL=0 -> YREF not below msl
c             IBMSL=1 -> YREF below msl
c     JS = Integer index incidcating at what index/level in array HTDUM()
c          YREF is located.
c     NEWL = New/adjusted number of levels to be stored in HREF() and REFREF()
c     RMU = Interpolated M-unit value at height YREF

      subroutine profref( yref, iflag )

      include 'tpem.inc'

      common / profwref / href(mxlvls), refref(mxlvls), nlvl
      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
     +                   profint(0:maxpts), ht(0:maxpts), is, lvlep

      nlvl = lvlep
      if( abs(yref) .gt. 1.e-3 ) then

         ibmsl = 0
```

```
         js = 0
         do i = 1, mxlvls
            href(i) = 0.
            refref(i) = 0.
         end do

c Check to see if reference height is below mean sea level.

         if( yref .lt. htdum(1) ) then
            ibmsl = 1
            js = 1

c Get refractivity profile level at which the height of the ground is just
c above. This level is JS.

         else
            nlvlm1 = nlvl - 1
            do i = 1, nlvlm1
               if(( yref .le. htdum(i+1) ) .and. ( yref .gt. htdum(i) ))
     +            js = i
            end do
         end if

c Determine the refractivity value at the ground and fill arrays HREF() and
c REFREF() with refractivity profile where height 0. now refers to the ground
c reference,i.e., either local ground height or HMINTER.

         if(( js .ne. 0 ) .or. ( ibmsl .eq. 1 )) then
            jsp1 = js + 1
            frac = (yref - htdum(js))/(htdum(jsp1) - htdum(js))
            rmu = refdum(js) + frac * (refdum(jsp1) - refdum(js))
            if( int( frac ) .eq. 1 ) js = jsp1
            newl = nlvl - js +1
            refref(1) = rmu
            href(1) = 0.
            k = js + 1
            do jk = 2, newl
               refref(jk) = refdum(k)
               href(jk) = htdum(k) - yref
               k = k + 1
            end do
            nlvl = newl
            if( iflag .eq. 0 ) then
               lvlep = nlvl
               do i = 1, mxlvls
                  refdum(i) = refref(i)
                  htdum(i) = href(i)
               end do
            end if
         end if
      else

c If the reference height is 0. then HREF() and REFREF() are equal.

         do i = 1, nlvl
            href(i) = htdum(i)
            refref(i) = refdum(i)
         end do
      end if

      end
```

8.1.13 Subroutine INTPROF

```
c ************************* SUBROUTINE INTPROF ******************************

c Module Name: INTPROF

c Module Security Classification: UNCLASSIFIED

c Purpose:  Performs a linear interpolation vertically with height on the
c           refractivity profile.  Stores interpolated profile in PROFINT().

c Version Number: 1.5

c INPUTS:
c   Argument List: NONE
c   Common: CON, HT(), HREF(), N, NLVL, REFREF()

c OUTPUTS:
c   Argument List: NONE
c   Common: PROFINT()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PEINIT, REFINTER

c Routines called: NONE

c GLOSSARY:
c   Input Variables:
c     For common variables refer to main glossary

c   Output Variables:
c     For common variables refer to main glossary

c   Local Variables:
c     HEIGHT = Height on which to interpolate
c     FRAC = Fractional height for interpolation

      SUBROUTINE intprof

      include 'tpem.inc'

      common / profwref / href(mxlvls), refref(mxlvls), nlvl
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
     +                   profint(0:maxpts), ht(0:maxpts), is, lvlep

      J=2

      DO I=0,N
         height = ht(i)
   40    IF((height .LE. href(J)) .OR. (J .GE. nlvl)) then
            k = j - 1
            FRAC = (height - href(k)) / (href(J) - href(k))
            profint(I) = (refref(k) + FRAC * (refref(J) -
     +                   refref(k))) * con
         else
            J=J+1
            GO TO 40
         end if
      end do

      END
```

8.2 Subroutine PESTEP

```
c *************************** SUBROUTINE PESTEP ******************************

c Module Name: PESTEP

c Module Security Classification: UNCLASSIFIED

c Purpose:  Propagates the field by one output range step DROUT.

c Version Number: 1.5

c INPUTS:
c   Argument list: HMINTER, RF(refractivity) structure,
c                  SV(systemvar) structure, TR(terrain) structure,
c                  VNP(inputvar) structure
c   Common: ALPHAV, C1, C2, C1M, C2M, DR, DR2, DROUT, DZ2, ENVPR(), FRSP(),
c           FTER, IG, N, RAV(), RK, ROOT, NM1, SLP(), U(), YLAST, YCUR, YCURM

c OUTPUTS:
c   Argument list: JEND, JSTART, MLOSS(), ROUT
c   Common:  ALPHAV, C1, C2, C1M, C2M, ENVPR(), IG, RAV(), RK, RNG, RNG2,
c            ROOT, U(), ULST(), YCUR, YCURM, YLAST

c Files Included: FFTSIZ.INC, TPEM.INC

c CALLING ROUTINES: MAIN DRIVER PROGRAM or TESS CSCI

c ROUTINES CALLED:  CALCLOS, DOSHIFT, FRSTP, GETALN, PHASE2, REFINTER

c GLOSSARY: For common variables, refer to main glossary

c  Input variables:
c    HMINTER = Minimum height of user-provided terrain profile.  This is
c              the height for which all internal calculations of the field
c              are referenced.
c    For structures RF, TR, VNP, SV refer to Glossary in PEINIT module.

c  Output variables:
c    JEND = Index at which the valid propagation loss values end.
c    JSTART = Index at which the valid propagation loss values begin.
c    MLOSS() = Array containing the propagation loss values in centibels,
c              at each output range point ROUT. All loss values returned
c              are referenced to height VNP.HMIN.
c    ROUT = Output range in meters.

c  Local variables:
c    KT = Counter for terrain profile.
c    R = Current PE range in meters.
c    RLAST = PE range at previous step in meters.
c    RMID = Range at which interpolation for range-dependent refractivity
c           profiles is performed.  This is equal to the range midway
c           between the current and next PE range.
c    SLOPE = Current slope of terrain segment.
c (Note: the following variables are only used for vertical polarization)
c    AR = Complex coefficient of partial linear solution to homogeneous equ.
c    BR = Complex coefficient of partial linear solution to homogeneous equ.
c    ARX = Partial linear solution to homogeneous equ.
c    BRX = Partial linear solution to homogeneous equ.
c    C1C = Summation argument in determining AR.
c    C2C = Summation argument in determining BR.
c    CD = RAV(i) or -RAV(i) depending on power index.
c    SUM1 = Summation term in determining AR.
c    SUM2 = Summation term in determining BR.
c    UI = U(i).
```

```
c     UNMI = U(n-i).
c     W() = Difference equation of complex PE field array.  Used in
c           intermediate calculations only for vertical polarization.
c     YM() = Particular solution of difference equation.  Used in
c            intermediate calculations only for vertical polarization.

      subroutine pestep( hminter, vnp, rf, tr, sv, ROUT, MLOSS, JSTART,
     +                   JEND )

      include 'tpem.inc'

      common / htvar / ylast, ycur, ycurm
      common / arrays / u(0:maxpts), filt(0:maxn4), frsp(0:maxpts),
     +                  envpr(0:maxpts), ulst(0:maxpts)
      common / rhstps / dr, drout, dzout, dr2, zout(mxzout)
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
      common / impedance / alphav, rav(0:maxpts), rng, rng2, c1, c2,
     +                     rk, c1m, c2m, ig, root

      record / inputvar / vnp
      record / refractivity / rf
      record / terrain / tr
      record / systemvar / sv

      logical fter

      complex c1, c2, rk, c1m, c2m, qi, ar, br, sum1, sum2, c1c, c2c
      complex alphav, rav, rng, rng2, ui, unmi, root, arx, brx, cd
      complex u, frsp, envpr, ulst, w(0:maxpts), ym(0:maxpts)

      integer*2 mloss(*)

      save r, kt, slope

      if( rout .le. 1.e-3 ) r = 0.
      rout = rout + drout

      DO while( r .lt. rout )

         if( r .gt. 0. ) ylast = ycur
         rlast = r

c Store the field arrays of the previous range step for subsequent horizontal
c interpolation at range ROUT.

         do i = 0, n
            ulst(i) = u(i)
         end do

         r = r + dr
         rmid = r - dr2

         if( fter ) then
            if( abs(r - dr) .le. 1.e-3 ) then
               slope = slp(1)
               kt = 1
            end if

c Check to see if current range is past a range point in terrain profile.
c If so, increment counter, determine terrain height at current range.

            do while((r .gt. tr.terx(kt+1)) .and. (kt .lt. tr.itp))
               kt = kt + 1
               slope = slp(kt)
```

```
            end do
            ycur = tr.tery(kt) + slope * ( r - tr.terx(kt) )

c Determine height at 1/2 range step - for interpolation on refractivity
c profiles.

            kp = kt
            do while( rmid .lt. tr.terx(kp) )
               kp = kp-1
            end do
            ycurm = tr.tery(kp) + slp(kp) * (rmid - tr.terx(kp))

c Calculate new complex refractive index and impedance term if using vertical
c polarization.

            if( sv.polar .eq. 'V' ) then
               if( r .gt. tr.rgrnd(ig+1) ) then
                  ig=ig + 1
                  call getaln( tr )
               end if
            end if

c Perform boundary shift for terrain case.

            if( slope .lt. 0. ) call doshift
         end if

         if( sv.polar .eq. 'V' ) then
            do i = 1, nm1
               w(i) = (u(i+1) - u(i-1)) / dz2 + alphav * u(i)
            end do
            call frstp( frsp, W )

c Propagate C1 and C2 coefficients to new range. NOTE: ONLY FOR SMOOTH
c SURFACE.

            c1 = c1 * c1m
            c2 = c2 * c2m
         else
            call frstp( frsp, U )
         end if

c If range-dependent and/or terrain case, then interpolate on profile.

         if(( rf.nprof .gt. 1 ) .or. ( fter )) then
           call refinter( rf, rmid, hminter )
           CALL PHASE2
         end if

c This follows steps 9-11 in Kuttler's formulation for vertical
c polarization. (Ref. viewgraphs from 1995 PE Workshop)

         if( sv.polar .eq. 'V' ) then
            ym(0) = cmplx(0.,0.)
            do i = 1, nm1
               ym(i) = dz2 * w(i) + root * ym(i-1)
            end do
            u(n) = cmplx(0.,0.)
            do i = 1, N
               nmi = n - i
               u(nmi) = root * (ym(nmi) - u(nmi+1))
            end do

            sum1 = cmplx( 0., 0. )
            sum2 = cmplx( 0., 0. )
            do i = 0, n
               nmi = n - i
```

```
            ui = u(i)
            unmi = u(nmi)
            if(( i .eq. 0 ) .or. (i .eq. n )) then
               ui = .5 * ui
               unmi = .5 * unmi
            end if

            iv = mod( i, 2 )
            cd = rav(i)
            if( iv .eq. 1 ) cd = -rav(i)

            c1c = ui * rav(i)
            c2c = unmi * cd

            sum1 = sum1 + c1c
            sum2 = sum2 + c2c
         end do

         ar = c1 - rk * sum1
         br = c2 - rk * sum2

         do i = 0, n
            arx = ar * rav(i)
            nmi = n - i
            iv = mod( nmi, 2 )
            cd = rav(nmi)
            if( iv .eq. 1 ) cd = -rav(nmi)
            brx = br * cd
            u(i) = u(i) + arx + brx
         end do
      end if

c Multiply by environment term.

      DO I = 1, nm1
         u(i) = u(i) * envpr(i)
      end do

c Perform boundary shift for terrain case.

      if(( fter ) .and. ( slope .ge. 0. )) call doshift

   end do

c Calculate propagation loss at range ROUT.

   call calclos( r, rout, rlast, vnp, hminter, MLOSS, JSTART, JEND )

   end
```

8.2.1 Subroutine DOSHIFT

```
c ************************ SUBROUTINE DOSHIFT ****************************

c Module Name: DOSHIFT

c Module Security Classification: UNCLASSIFIED

c Purpose: Shifts the field by the # of bins corresponding to height of
c          the ground.

c Version Number: 1.5

c INPUTS:
c    Argument List: NONE
c    Common: DELZ, N, NM1, U(), YCUR, YLAST

c OUTPUTS:
c    Argument List: NONE
c    Common: U()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PESTEP

c Routines called: NONE

c GLOSSARY: For common variables refer to main glossary

c   Input Variables:
c     See main glossary for all common variable definitions

c   Output Variables:
c     U() = Complex PE field array containing newly shifted field solution.

c   Local Variables:
c     INCR = Integer indicating which direction to shift field U().
c            INCR = 1 -> terrain slope is positive, shift down.
c            INCR = -1 -> terrain slope is negative, shift up.
c     JEND = End index in U() at which to end shifting.
c     JST = Start of index in U() at which to begin shifting
c     KBIN = # of bins to shift field.
c     YDIF = Height difference between current and last ground elevation.

      subroutine doshift

      include 'tpem.inc'

      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / htvar / ylast, ycur, ycurm
      common / arrays / u(0:maxpts), filt(0:maxn4), frsp(0:maxpts),
     +                  envpr(0:maxpts), ulst(0:maxpts)

      complex u, frsp, envpr, ulst

      ydif = ycur - ylast
      kbin = nint( abs(ydif) / delz )
      if( kbin .eq. 0 ) return

c If slope is positive then shift array elements down.

      if( ydif .ge. 0. ) then
         incr = 1
         jst = 1
         jend = nm1 - kbin
```

```
      else

c If slope is negative then shift array elements up.

         incr = -1
         jst = nm1
         jend = kbin + 1
      end if

      kinc = incr * kbin
      do j = jst, jend, incr
         jk = j + kinc
         u(j) = u(jk)
      end do

c If shifted down, fill the upper KBIN elements of U() with zero.
c If shifted up, fill the lower KBIN elements of U() with zero.

      if( incr .gt. 0 ) then
         nst = n - kbin
         do j = nst, nm1
            u(j) = 0.
         end do
      else
         do j = 1, kbin
            u(j) = 0.
         end do
      end if

      end
```

8.2.2 Subroutine GETALN

```
c ************************** SUBROUTINE GETALN ******************************

c Module Name: GETALN

c Module Security Classification: UNCLASSIFIED

c Purpose:  Computes the impedance term ALPHAV and the complex index of
c           refraction for finite conductivity and vertical polarization
c           calculations. These formulas follow Kuttler's method. (Ref.
c           Kuttler's viewgraphs from PE modeler's workshop).

c Version Number: 1.5

c INPUTS:
c   Argument List: TR structure
c   Common:  DELZ, DR, FKO, N, QI, WL
c   Parameter: PI

c OUTPUTS:
c   Argument List: NONE
c   Common: ALPHAV, C1, C2, C1M, C2M, IG, RAV(), RK, RNG, RNG2, ROOT

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PESTEP, XYINIT

c Routines Called: NONE

c GLOSSARY:  For common variables, refer to main glossary.  For parameters,
c            refer to TPEM.INC

c   Input Variables:
c     TR = Terrain structure for external terrain data elements.
c       TR.DIELEC(,) = 2-dimensional array containing the relative
c                      permittivity and conductivity; DIELEC(1,i) and
c                      DIELEC(2,i), respectively. Only needs to be specified
c                      if using IGRND(i) = 5, otherwise, TPEM will
c                      calculate based on frequency and ground types 0-4.

c   Output Variables:
c     Refer to main glossary for all common variables.

c   Local Variables:
c     A = Exponential term in computing C1M and C2M
c     AD = Dummy complex variable used to in calculating complex root ROOT
c          and C1M and C2M
c     R2 = Third element of RAV(), i.e., ROOT**2
c     RAVLN = Complex natural logarithim of ROOT
c     R2N = (N+1)th element of RAV() squared, i.e., (ROOT**N)**2
c     S1 = Imaginary term in complex index of refraction squared, RNG2
c     SQRAD = Complex square root of 1+AD**2

      subroutine getaln( tr )

      include 'tpem.inc'

      common / rhstps / dr, drout, dzout, dr2, zout(mxzout)
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / impedance / alphav, rav(0:maxpts), rng, rng2, c1, c2, rk,
     +                     c1m, c2m, ig, root
      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
```

```
      record / terrain / tr

      complex alphav, rav, rng, rng2, c1, c2, rk, c1m, c2m, root
      complex ad, sqrad, r2, a, ravln, qi, r2n

      logical fter

      s1 = tr.dielec(2,ig) * 60. * wl
      rng2 = cmplx( tr.dielec(1,ig), s1 )
      rng = csqrt( rng2 )
      alphav = qi * fko / rng
      ad = alphav * delz

      sqrad = csqrt( 1. + ad**2 )

c Root for vertical polarization only.

      root = sqrad - ad
      do i = 0, n
         rav(i) = root**i
      end do

      r2 = rav(2)
      r2n = rav(n)**2
      rk = 2.*(1. - r2) / (1. + r2) / (1. - r2n)
      a = dr * qi / 2. / fko
      ravln = clog( root )
      ad = (ravln / delz)**2
      c1m = cexp( a * ad )
      ad = ( (ravln - qi * pi ) / delz )**2
      c2m = cexp( a * ad )

      end
```

8.2.3 Subroutine FRSTP

```
c ************************* SUBROUTINE FRSTP *****************************

c Module Name: FRSTP

c Module Security Classification: UNCLASSIFIED

c Purpose: Propagates the field FARRAY() in free space by one range step.

c Version Number: 1.5

c INPUTS:
c   Argument List: FARRAY(), FRSP()
c   Common: NM1

c OUTPUTS:
c   Argument List: FARRAY()
c   Common: NONE

c Files Included: NONE

c Calling Routines: PESTEP

c Routines Called: FFT

c GLOSSARY: For common variables refer to main glossary

c   Input Variables:
c     FARRAY() = Field array to be propagated one range step in free space
c                (z-space). If polarization is horizontal, then upon entry
c                FARRAY() is the field array U().  If using vertical
c                polarization, FARRAY() is W().
c     FRSP() = Complex free space propagator term.

c   Output Variables:
c     FARRAY() = Field array propagated one range step in free space
c                (z-space). If polarization is horizontal, then upon exit
c                FARRAY() is the field array U().  If using vertical
c                polarization, FARRAY() is W().

      subroutine frstp( frsp, FARRAY )

      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1

      complex frsp(0:*), farray(0:*)

      call fft( farray )    !Transform to Fourier space

      DO I = 1, NM1                !Multiply by free-space propagator
         farray(i) = farray(i) * frsp(i)
      end do

      call fft( farray )    !Transform back to z-space

      end
```

8.2.4 Subroutine REFINTER

```
c ************************* SUBROUTINE REFINTER **************************

c Module Name: REFINTER

c Module Security Classification: UNCLASSIFIED

c Purpose: Interpolates vertically and horizontally on the refractivity
c          profiles.

c Version Number: 1.5

c INPUTS:
c   Argument List: HMINTER, RANGE, RF structure
c   Common: IS, RV2, YCURM

c OUTPUTS:
c   Argument List: NONE
c   Common:  IS, LVLEP, PROFINT(), RV2

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PESTEP

c Routines called: INTPROF, PROFREF, REMDUP

c GLOSSARY: For common variables refer to main glossary

c   Input Variables:
c     HMINTER = Internal reference height in meters (minimum height of terrain
c               profile).
c     RANGE = Range for profile interpolation.
c     RF = Refractivity structure for external environmental data elements.
c       RF.LVLEP = Number of levels in refractivity profile (for range
c                  dependent case all profiles must have same number of
c                  levels).
c       RF.REFMSL(,) = 2-dimensional array containing refractivity with
c                      respect to mean sea level of each profile.  Array
c                      format must be REFMSL(I,J) = M-unit value at Ith
c                      level of Jth profile.  J = 1 for range-independent
c                      cases.
c       RF.HMSL(,) = 2-dimensional array containing heights in meters with
c                    respect to mean sea level of each profile.  Array format
c                    must be HMSL(I,J) = height of Ith level of Jth profile.
c                    J = 1 for range-independent cases.
c       RF.RNGPROF() = Ranges of each profile in meters, i.e., RNGPROF(I) =
c                    range of Ith profile.  RNGPROF(1) should always be
c                    equal to 0.
c       RF.NPROF = number of profiles.  Equals 1 for range-independent cases.

c   Output Variables:
c       For common variables refer to main glossary

c   Local Variables:
c     FV = Fractional range over which to interpolate
c     J = index of last refractivity profile (for range-dependent case).
c     RV1 = range of last refractivity profile (for range-dependent case).

      subroutine refinter( rf, range, hminter )

      include 'tpem.inc'

      common / htvar / ylast, ycur, ycurm
      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
```

```
     +                   profint(0:maxpts), ht(0:maxpts), is, lvlep
      record / refractivity / rf

      save j, rvl
      data j, rvl / 0, 0. /

c One-line interpolation function

      pint( p1, p2 ) = p1 + fv * ( p2 - p1 )

      lvlep = rf.lvlep

c If there is a range-dependent refractivity profile then interpolate
c horizontally using the two surrounding profiles at range RANGE with all
c duplicate levels.

      if( rf.nprof .gt. 1 ) then
         IF( range .gt. rv2 ) then
            j = is
            IS=IS+1
            rv1=rv2
            rv2=rf.rngprof(IS)
         end if

         FV=(range-rv1)/(rv2-rv1)

         do i = 1, lvlep
            refdum(i) = pint( rf.refmsl(i,j), rf.refmsl(i,is) )
            htdum(i) = pint( rf.hmsl(i,j), rf.hmsl(i,is) )
         end do

c Now remove all duplicate levels with LVLEP now being the # of points in the
c profile at range RANGE.

         call remdup
         call profref( hminter, 0 )

c At this point REFDUM() and HTDUM() are referenced to HMINTER.

      end if

c Using BS method must determine height and M-unit profiles relative to
c ground, where YCURM is now the height of the local ground above the
c reference height HMINTER.

      call profref( ycurm, 1 )

c Interpolate vertically with height.  PROFINT() is now an N-point (N=2**NFFT)
c array containing the interpolated M-unit values for the refractivity at
c range RANGE.

      call intprof

      end
```

8.2.5 Subroutine REMDUP

```
c ************************ SUBROUTINE REMDUP ****************************

c Module Name: REMDUP

c Module Security Classification: UNCLASSIFIED

c Purpose: Removes duplicate refractivity levels in profile.

c Version Number: 1.5

c INPUTS:
c   Argument List: NONE
c   Common: HTDUM(), LVLEP, REFDUM()

c OUTPUTS:
c   Argument List: NONE
c   Common: HTDUM(), LVLEP, REFDUM()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: REFINIT, REFINTER

c Routines called: NONE

c GLOSSARY:

c   Input Variables:
c     For common variables refer to main glossary

c   Output Variables:
c     For common variables refer to main glossary

c   Local Variables:
c     HT1 = Height at (I)th refractivity level
c     HT2 = Height at (I+1)th refractivity level

      subroutine remdup

      include 'tpem.inc'

      common / parinit / rv2, refdum(mxlvls), htdum(mxlvls),
     +                   profint(0:maxpts), ht(0:maxpts), is, lvlep

c  Remove all duplicate levels in interpolated profile

      i = 1
      do while( i .lt. lvlep )
         ht1 = htdum(i)
         ht2 = htdum(i+1)
         if( abs(ht1-ht2) .le. 1.e-3 ) then
            lvlep = lvlep - 1
            do j = i, lvlep
               jp1 = j + 1
               htdum(j) = htdum(jp1)
               refdum(j) = refdum(jp1)
            end do
            i = i - 1
         end if
         i = i + 1
      end do

      end
```

8.2.6 Subroutine CALCLOS

```
c **************************** SUBROUTINE CALCLOS ***************************

c Module Name: CALCLOS

c Module Security Classification: UNCLASSIFIED

c Purpose: Determines the propagation loss at each output range ROUT and
c          all heights up to ZLIM.

c Version Number: 1.5

c INPUTS:
c    Argument List: HMINTER, R, RLAST, ROUT, VNP structure
c    Common: DR, DROUT, DZOUT, FTER, HLIM(), HMREF, PLCNST, RPE, U(), ULST(),
c            YLAST, YCUR, ZLIM, ZOUT()
c    Parameter: MXZOUT

c OUTPUTS:
c    Argument List: JEND, JSTART, MLOSS()

c Files Included: FFTSIZ.INC, TPEM.INC

c Calling Routines: PESTEP

c Routines Called: function GETPFAC

c GLOSSARY:  For common variables refer to main glossary.  For parameters
c            refer to TPEM.INC

c    Input Variables:
c      HMINTER = Reference height for internal calculations of the field U().
c      R = PE range in meters.
c      RLAST = Last PE range in meters.
c      ROUT = Output range in meters.

c    Output Variables:
c      JEND = Index at which valid loss values in MLOSS ends.
c      JSTART = Index at which valid loss values in MLOSS begin.
c      MLOSS() = 2 byte integer array containing propagation loss values in
c                centibels.

c    Local Variables:
c      FSLROUT = Free space loss at ROUT
c      IC = Counter for HLIM() array
c      IP1 = Index in array RFAC1() corresponding to ground height at
c            previous PE range.  All array elements in RFAC1() from 1 to
c            IP1 are set equal to PFACMIN.
c      IP2 = Index in array RFAC2() corresponding to ground height at
c            current PE range.  All array elements in RFAC2() from 1 to
c            IP2 are set equal to PFACMIN.
c      IZG = Number of height output points corresponding to local ground
c            height at current output range ROUT, i.e., IZG*DZOUT = ZINT.
c      PFACMIN = Minimum propagation factor allowed to avoid overflow
c      RFAC1() = Array of propagation factor at valid output height points
c                for range RLAST
c      RFAC2() = Array of propagation factor at valid output height points
c                for range R.
c      RLOG = 10. times the logarithm (base 10) of the current PE range R
c      RLOGLST = 10. times the logarithm (base 10) of the last PE range RLAST
c      RLOSS = Real propagation loss in cB
c      XX = Fractional range at which to interpolate propagation factor.
c      YCH = Height of terrain at current range step relative to reference
c            height HMREF.
```

```
c      YCT = Height of terrain at current range step relative to minimum
c            terrain height HMINTER.
c      YLH = Height of terrain at last range step relative to reference
c            height HMREF.
c      YLT = Height of terrain at last range step relative to minimum
c            terrain height HMINTER.
c      ZEND2 = Height at which to stop calculating propagation factor
c      ZHT = Height of desired output point relative to HMINTER
c      ZINT = Interpolated ground height at current output range ROUT.

      subroutine calclos( r, rout, rlast, vnp, hminter, MLOSS, JSTART,
     +                    JEND )

      include 'tpem.inc'

      common / miscvar / fnorm, cnst, delp, thetamax, plcnst, qi,
     +                   antref, rpe, hlim(mxrout), slp(mxter), fter,
     +                   hmref, htlim
      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1
      common / rhstps / dr, drout, dzout, dr2, zout(mxzout)
      common / arrays / u(0:maxpts), filt(0:maxn4), frsp(0:maxpts),
     +                  envpr(0:maxpts), ulst(0:maxpts)
      common / htvar / ylast, ycur, ycurm
      common / trvar / dmdh(mxlvls), zlim, jls, thetalaunch, rlim

      record / inputvar / vnp

      complex u, frsp, envpr, ulst, qi

      integer*2 mloss(*)

      logical fter

      dimension rfac1(mxzout), rfac2(mxzout)

      save ic

      data pfacmin / 300. /      !Set minimum propagation factor of 300 dB

c Define in-line function for linear interpolation.

      plint(pl1, pl2, frac) = pl1 + frac * ( pl2 - pl1 )

c Initialize counter for HLIM array.

      if( abs(rout - drout) .le. 1.e-3 ) ic = 1

      ych = ycur - hmref
      yct = ycur + hminter
      ylh = ylast - hmref
      ylt = ylast + hminter

c Get height of ground at output range ROUT and determine number of vertical
c output points that correspond to the ground height.  Fill the loss array
c MLOSS with zeros to represent ground for those vertical output points.

      xx = (rout - rlast) / dr
      zint = plint( ylast, ycur, xx )
      izg = int( (zint-hmref) / dzout  )
      do i = 1, izg
         mloss(i) = 0
      end do

      jstart = amax0( 1, izg+1 )

      if( rout .gt. rpe ) then
```

```fortran
c If current output range is greater than RPE then begin calculation of loss
c values and return them in MLOSS().

        rloglst = 0.
        if( rlast .gt. 0. ) rloglst = 10. * alog10( rlast )
        rlog = 10. * alog10( r )
        fslrout = 20. * alog10(rout) + plcnst       !free space loss at ROUT

c Determine values of array elements corresponding to the ground and set these
c to the minimum propagation factor (-300) for later interpolation.

        if( fter ) then
           ip1 = int( ylh / dzout )
           ip2 = int( ych / dzout )
           ip1 = amax0( 0, ip1 )
           ip2 = amax0( 0, ip2 )

           do i = 1, ip1
              rfac1(i) = pfacmin
           end do
           do i = 1, ip2
              rfac2(i) = pfacmin
           end do
           ip1 = ip1 + 1
           ip2 = ip2 + 1

        else
           ip1 = 1
           ip2 = 1
        end if

c Determine height/integer value at which to stop calculating loss.
c NOTE: For terrain cases, ray tracing was performed
c using the direct ray angle and sometimes HLIM(i) may be less than the
c local ground height. The GOTO statement is used just as a safety factor
c in this case.

        zend1 = amax1( zint, hlim(ic) )
        zend2 = amin1( zlim, zend1 )
        jend = amax0( 0, nint( (zend2-hmref) / dzout ) )

        if( jend .lt. jstart ) goto 5

c Get propagation factor at valid heights from field at previous PE range step.

        if( rloglst .gt. 0. ) then
           do i = ip1, jend
              zht = zout(i) - ylt
              rfac1(i) = getpfac( ulst, rloglst, zht )
           end do
        end if

c Get propagation factor at valid heights from field at current range step.

        do i = ip2, jend
           zht = zout(i) - yct
           rfac2(i) = getpfac( u, rlog, zht )
        end do

c Interpolate between the two PE range steps to get propagation loss at range
c ROUT.

        do k = jstart, jend
           if( rloglst .gt. 0. ) then
              rloss = 10.*( plint( rfac1(k), rfac2(k), xx ) + fslrout )
              mloss(k) = int2( rloss )
           else
```

```
              mloss(k) = int2( 10. * ( rfac2(k) + fslrout ) )
           end if
        end do

 5      continue

c Fill remainder of array with -1 indicating non-valid loss values.

        jn = jend + 1
        do i = jn, vnp.nzout
           mloss(i) = -1
        end do

      else

c If current output range is less than RPE then there are no current valid
c loss values at any height - fill MLOSS with -1. JSTART and JEND will be
c equal and will have a value of 1 if smooth surface case, otherwise will
c have a value of the nearest integer multiple of DZOUT corresponding to the
c height of the local ground.

        jend = jstart
        do i = jstart, vnp.nzout
           mloss(i) = -1
        end do

      end if

      ic = ic + 1

      end
```

8.2.7 Function GETPFAC

```
c ***************************** FUNCTION GETPFAC ****************************

c Module Name: GETPFAC

c Module Security Classification: UNCLASSIFIED

c Purpose:  Performs linear interpolation in height on the power and then
c           calculates propagation factor in dB.

c Version Number: 1.5

c INPUTS:
c   Argument List: HEIGHT, RLOG, U()
c   Common: DELZ

c OUTPUTS:
c   Function: GETPFAC

c Files Included: NONE

c Calling Routines: CALCLOS

c Routines Called: NONE

c GLOSSARY: For common variables refer to main glossary

c   Input Variables:
c     HEIGHT = receiver height in meters.
c     RLOG = 10. * logarithm (base 10) of PE range.
c     U() = Complex field array.

c   Output Variables:
c     GETPFAC = Propagation factor in dB at specified height.

c   Local Variables:
c     FB = Real (fractional) number of bin widths corresponding to desired
c          receiver height
c     FR = Fractional bin width on which to interpolate
c     NB = Integer number of bin widths for which the desired receiver
c          height is just greater than.
c     NBP1 = Integer number of bin widths for which the desired receiver
c            height is just below (i.e., NB+1)
c     POW0 = Magnitude of field at height immediately below desired height.
c     POW1 = Magnitude of field at height immediately above desired height.
c     POWMIN = Minimum field magnitude allowed - used underflow/overflow
c              problems.
c     RPOW = Interpolated field magnitude
c     U0 = Complex field at bin directly below (NB) desired receiver height
c     U1 = Complex field at bin directly above (NBP1) desired receiver height

      function GETPFAC( u, rlog, height )

      common / pevar / wl, fko, delz, n, ln, zmax, n34, con, dz2, nm1

      complex u(0:*), u0, u1

      data powmin/1.e-13/

      fb = height / delz
      nb=int(fb)
      fr=fb-float(nb)
      nbp1=nb+1
```

```
u0=u(nb)
u1=u(nbp1)

pow0 = cabs( u0 )
pow1 = cabs( u1 )

pow = pow0 + fr * (pow1 - pow0)

rpow = amax1( pow, powmin )
getpfac = -20.*alog10( rpow ) - rlog

end
```

8.3 INCLUDE Interface Source Code

8.3.1 FFTSIZ.INC

```
c MXNFFT:  Maximum power of 2 for transform size
c MAXPTS:  Maximum size of arrays for the real and imaginary fields

      integer*4 maxpts, mxnfft
      parameter ( mxnfft = 14)
      parameter ( maxpts = 2**mxnfft )
```

8.3.2 TPEM.INC

```
include 'fftsiz.inc'

      integer*4 maxn4, mxzout, mxrout, mxlvls, mxnprof, mxter

      parameter ( maxn4 = maxpts/4 )  !used for filter array - filters
                                      !upper 1/4 of field.
      parameter ( pi = 3.1415926 )  !Self-explanatory
      parameter ( mxzout = 385 )  !Maximum number of output height points
      parameter ( mxrout = 440 )  !Maximum number of output range points
      parameter ( mxlvls = 300 )  !Maximum number of height/M-unit levels
      parameter ( mxnprof = 30 )  !Maximum number of profiles allowed for
                                  !range-dependent environment.
      parameter ( mxter = 1002 )  !Maximum number of height/range points
                                  !allowed for terrain profile

c ERRORFLAG:
c  LERR6 =  Logical flag that allows for greater flexibility in allowing error
c           -6 to be bypassed.  If set to .TRUE. then trapping for this error
c           occurs, otherwise it can be totally ignored by main driver
program.
c           (Within the TPEM program it is handled as a warning).  If this
c           error is bypassed (LERR6 = .FALSE.) terrain profile is extended to
c           RMAX with same elevation height of last valid terrain profile
point.
c  LERR12 = Same as LERR6 - allows for trapping of this error.  If LERR12 =
c           .FALSE., then (for range-dependent case) if range of last
c           refractivity profile entered is less than RMAX, the environment
c           is treated as homogeneous from the last profile entered to RMAX.

      structure / errorflag /
         logical lerr6
         logical lerr12
      end structure

c INPUTVAR:
c   HMAX = maximum output height with respect to m.s.l. in meters
c   HMIN = minimum output height with respect to m.s.l. in meters
c   RMAX = maximum output range in meters
c   NZOUT = integer number of output height points desired
c   NROUT = integer number of output range points desired
c   PROPANG = Maximum problem (propagation) angle in degrees desired for
c             solution. If set to 0., then TPEM will determine it's own.

      structure / inputvar /
         real hmax
         real hmin
         real rmax
         integer*4 nzout
         integer*4 nrout
```

```
         real propang
      end structure

c REFRACTIVITY:
c   LVLEP = number of levels in refractivity profile (for range dependent case
c           all profiles must have same number of levels)
c   REFMSL() = 2-dimensional array containing refractivity with respect to
mean
c              sea level of each profile.  Array format must be REFMSL(I,J) =
c              M-unit value at Ith level of Jth profile.  J = 1 for range-
c              independent cases.
c   HMSL() = 2-dimensional array containing heights in meters with respect to
c            mean sea level of each profile.  Array format must be HMSL(I,J) =
c            height of Ith level of Jth profile.  J = 1 for range-independent
c            cases.
c   RNGPROF() = ranges of each profile in meters, i.e., RNGPROF(I) = range of
c               Ith profile.  RNGPROF(1) should always be equal to 0.
c   NPROF = number of profiles.  Equals 1 for range-independent cases.

      structure / refractivity /
         integer*4 lvlep
         real refmsl(mxlvls, mxnprof)
         real hmsl(mxlvls, mxnprof)
         real rngprof(mxnprof)
         integer*4 nprof
      end structure

c SYSTEMVAR:
c   FREQ = frequency in MHz
c   ANTHT = transmitting antenna height above local ground in meters.
c   BWIDTH = half-power (3 dB) antenna pattern beamwidth in degrees (.5 to
45.)
c   ELEV = antenna pattern elevation angle in degrees. (-10 to 10)
c   POLAR = 1-character string indicating polarization.  H-horizontal,
c           V-vertical
c   IPAT = integer value indicating type of antenna pattern desired
c          IPAT = 0 -> omni
c          IPAT = 1 -> gaussian
c          IPAT = 2 -> sinc x
c          IPAT = 3 -> csc**2 x
c          IPAT = 4 -> generic height-finder

      structure / systemvar /
         real freq
         real antht
         real bwidth
         real elev
         character*1 polar
         integer*4 ipat
      end structure

c TERRAIN:
c  TERX() = range points of terrain profile in meters
c  TERY() = height points of terrain profile in meters
c  ITP = number of height/range pairs in profile
c  IGR = number of different ground types specified
c  IGRND() = type of ground composition for given terrain profile - can vary
c            with range.  Different ground types are: 0 = sea water,
c            1 = fresh water, 2 = wet ground, 3 = medium dry ground,
c            4 = very dry ground, 5 = user defined (in which case, values of
c            relative permittivity and conductivity must be given).
c  RGRND() = ranges at which the ground types apply
c  DIELEC(,) = 2-dimensional array containing the relative permittivity and
c              conductivity; DIELEC(1,i) and DIELEC(2,i), respectively.
c              Only needs to be specified if using IGRND(i) = 5, otherwise,
c              TPEM will calculate based on frequency and ground types 0-4.
```

```
structure / terrain /
   real terx(mxter)
   real tery(mxter)
   integer*4 itp
   integer*4 igr
   integer*4 igrnd(50)
   real rgrnd(50)
   real dielec(2,50)
end structure
```

SOFTWARE TEST DESCRIPTION

FOR THE

TERRAIN PARABOLIC EQUATION MODEL CSCI

May 1, 1997

Prepared for:

Space and Naval Warfare Systems Command (PMW-185)
Washington, DC

and

Naval Sea Systems Command (PEO USW ASTO-E/F)
Washington, DC

Prepared by:

Naval Command, Control and Ocean Surveillance Center
Research, Development, Test and Evaluation Division
Tropospheric Branch (Code D883)
San Diego, CA 92152-738

TABLE OF CONTENTS

LIST OF TABLES

1. SCOPE

1.1 Identification.

Terrain Parabolic Equation Model (TPEM) computer software configuration item (CSCI). The purpose of the TPEM CSCI is to calculate range-dependent electromagnetic (EM) system propagation loss within a heterogeneous atmospheric medium over variable terrain, where the radio-frequency index of refraction is allowed to vary both vertically and horizontally. Numerous Tactical Environmental Support System (TESS) applications require EM-system propagation loss values. The TPEM model described by this document may be applied to two such TESS applications, one which displays propagation loss on a range versus height scale (commonly referred to as a coverage diagram) and one which displays propagation loss on a propagation loss versus range/height scale (commonly referred to as a loss diagram).

1.2 Document Overview.

This document specifies the test cases and test procedures necessary to perform qualification testing of the TPEM CSCI. A discussion of precise input values of each input variable required to perform the test together with final expected test results is presented.

2. REFERENCE DOCUMENTS

(a) Commander-In-Chief, Pacific Fleet Meteorological Requirement (PAC MET) 87-04, "Range Dependent Electromagnetic Propagation Models."

(b) Naval Oceanographic Office, "Software Documentation Standards and Coding Requirements for Environmental System Product Development," April 1990.

(c) Naval Command, Control and Ocean Surveillance Center; Research, Development, Test and Evaluation Division (NRaD), "Operational Concept Document for the Terrain Parabolic Equation Model CSCI", Apr-97.

(d) Naval Command, Control and Ocean Surveillance Center; Research, Development, Test and Evaluation Division (NRaD), "Software Requirements Specification for the Terrain Parabolic Equation Model CSCI," Apr-97

(e) Naval Command, Control and Ocean Surveillance Center; Research, Development, Test and Evaluation Division (NRaD), "Software Design Document for the Terrain Parabolic Equation Model CSCI," Apr-97.

(f) Barrios, A. E., "Terrain Parabolic Equation Model (TPEM) Version 1.5 User's Manual," Naval Command, Control and Ocean Surveillance Center, RDT&E Division, San Diego, CA, NRaD TD 2898, February 1996.

3. TEST PREPARATIONS

3.1 Hardware Preparation

Not applicable

3.2 Software Preparation

A short driver program, MAIN.FOR, has been provided in Section 7. This program calls the main software units, PEINIT CSC and PESTEP CSC, that comprise the TPEM CSCI. The driver program listed in Appendix A is shown only to demonstrate how one would access the TPEM CSCI and to exercise the test cases listed in the following section. It has been written to read in all necessary input data for the following test cases from files in a specific format. All necessary input information is presented in tabular form in section 4.1.1 and the input files for each test case are listed in Appendix B. Ultimately, it is the responsibility of the TESS CSCI application designer to provide the necessary input in the form required by the TPEM CSCI.

Certain parameter values must be specified in the include files TPEM.INC and FFTSIZ.INC. The values of the external implementation constants declared as parameters and specified within the include files are listed in Table 3-1. The values are the same for all 14 tests and are not explicitly listed for each separate test case.

Table 3-1 External Implementation Constants Declared Within the TPEM.INC and FFTSIZ.INC

Name	Value
mxnfft	14
maxpts	16384
mxlvls	300
mxrout	440
mxzout	385
mxnprof	30
mxter	1002

3.3 Other Pretest Preparation.

None.

4. TEST DESCRIPTIONS

The test specification for the TPEM CSCI consists of 14 separate tests that have been selected to exercise all subroutines and functions of the code. For ease of testing, each of these 14 tests has been given a name describing which portion of the TPEM CSCI is being exercised. All 14 tests and their descriptions are listed in Table 4-1.

Table 4-1 Test Names and Descriptions

Test Name	Description
BLOCK	Tests the TPEM CSCI when the terrain profile consists of a vertical flat-topped block or obstacle in which the terrain slope is undefined.
COSEC2	Tests the TPEM CSCI when antenna pattern is of cosecant-squared type.
EDUCT	Tests the TPEM CSCI when the refractivity consists of a 14 meter evaporation duct profile.
GAUSS	Tests the TPEM CSCI when antenna pattern is of Gaussian type.
HILL	Tests the TPEM CSCI when the terrain profile consists of a rounded hill.
HORZ	Tests the TPEM CSCI for horizontal polarization antenna and standard atmosphere.
HTFIND	Tests the TPEM CSCI when antenna pattern is of generic height-finder type.
MIDEL	Tests the TPEM CSCI for Gaussian type antenna pattern and non-zero elevation angle.
RDLONGB	Tests the TPEM CSCI for range-dependent refractivity over a DTED-extracted terrain profile from Long Beach to Point Mugu.
RNGDEP	Tests the TPEM CSCI for range-dependent refractivity over smooth earth (over-water case).
SINEX	Tests the TPEM CSCI when antenna pattern is of Sine(X)/X type.
VERTSEA	Tests the TPEM CSCI for vertical polarization antenna over smooth earth (over-water case).
VERTMIX	Tests the TPEM CSCI for vertical polarization antenna over mixed land-sea terrain path.
WEDGE	Tests the TPEM CSCI when the terrain profile consists of a triangular wedge.

4.1 Requirements Addressed

Not applicable.

4.2 Prerequisite Conditions

None.

4.3 Test Inputs

Variable names that are part of structure elements are listed with their respective variable structure names as it appears in MAIN.FOR via the RECORD statement. For instance, variables comprising the *systemvar* structure are listed as *sv.freq*, *sv.antht*, etc., where *sv* is the variable name assigned to the structure of type *systemvar* and specific data element names correspond to the names used within reference e. Also, although there are actual values for all input parameters listed in the input files in Appendix B, some are ignored depending on the values of certain input parameters. Those input parameters that are inapplicable depending on the test case are listed as "N/A" in the tables.

The external environmental data element requirements are listed in Table 4-2 for each test name, with Table 4-3 through Table 4-7 providing specific height and M-unit values. The external EM system data element requirements are listed in Table 4-8.

Table 4-2 External Environmental Data Element Requirements

Test Name	*rf.hmsl* Table	*rf.refmsl* Table	*rf.nprof*	*rf.lvlep*	*rf.rngprof* meters
BLOCK	4-3	4-3	1	2	0.
COSEC2	4-3	4-3	1	2	0.
EDUCT	4-5	4-5	1	21	0.
GAUSS	4-3	4-3	1	2	0.
HILL	4-3	4-3	1	2	0.
HORZ	4-3	4-3	1	2	0.
HTFIND	4-3	4-3	1	2	0.
MIDEL	4-3	4-3	1	2	0.
RDLONGB	4-6	4-6	2	4	Table 4-6
RNGDEP	4-7	4-7	2	4	Table 4-7
SINEX	4-3	4-3	1	2	0.
VERTSEA	4-4	4-4	1	4	0.
VERTMIX	4-3	4-3	1	2	0.
WEDGE	4-3	4-3	1	2	0.

Table 4-3 Standard Atmosphere with 118 M/km Gradient

i	*rf.hmsl(i,1)* meters	*rf.refmsl(i,1)* M-unit
1	0.	0.
2	1000.	118.

Table 4-4 300 Meter Surface Based Duct Atmosphere

i	*rf.hmsl(i,1)* meters	*rf.refmsl(i,1)* M-unit
1	0.	339.0
2	250.	368.5
3	300.	319.0
4	1000.	401.6

Table 4-5 Atmosphere with 14 Meter Evaporation Duct

i	*rf.hmsl(i,1)* meters	*rf.refmsl(i,1)* M-unit
1	0.000	339.00
2	0.040	335.10
3	0.100	333.66
4	0.200	332.60
5	0.398	331.54
6	0.794	330.51
7	1.585	329.53
8	4.362	328.65
9	6.310	327.96
10	12.589	327.68
11	14.000	327.67
12	25.119	328.13
13	39.811	329.25
14	50.119	330.18
15	63.096	331.44
16	79.433	334.32
17	100.000	335.33
18	125.893	338.20
19	158.489	341.92
20	199.526	346.69
21	209.526	347.87

Table 4-6 Range-dependent Atmosphere, Standard Atmosphere to low elevated Duct

	Standard Atmosphere *rf.rngprof(1)* = 0. km		Low Elevated Duct *rf.rngprof(2)* = 100 km	
i	*rf.hmsl(i,1)* meters	*rf.refmsl(i,1)* M-unit	*rf.hmsl(i,2)* meters	*rf.refmsl(i,2)* M-unit
1	0.	0.	0.	330.
2	0.	0.	191.	352.5
3	0.	0.	201.	343.3
4	1000.	118.	1201.	461.1

Table 4-7 Range-dependent Atmosphere, Low Elevated Duct to High Elevated Duct

	Low Elevated Duct *rf.rngprof(1)* = 0. km		High Elevated Duct *rf.rngprof(2)* = 250. km	
i	*rf.hmsl(i,1)* meters	*rf.refmsl(i,1)* M-unit	*rf.hmsl(i,2)* meters	*rf.refmsl(i,2)* M-unit
1	0.	330.	0.	330.
2	100.	342.5	600.	405.
3	230.	312.5	730.	375.
4	2000.	517.8	2000.	522.3

Table 4-8 External EM System Data Element Requirements.

Test Name	*sv.freq* MHz	*sv.antht* meters	*sv.ipat* note a	*sv.polar*	*sv.bwidth* deg	*sv.elev* deg
BLOCK	1000.	25.	0	H	N/A	N/A
COSEC2	1000.	25.	3	H	1.	0.
EDUCT	10000.	15.	1	H	5.	0.
GAUSS	1000.	25.	1	H	1.	0.
HILL	1000.	25.	0	H	N/A	N/A
HORZ	1000.	25.	0	H	N/A	N/A
HTFIND	1000.	25.	4	H	2.	0.
MIDEL	1000.	10.	1	H	1.	5.
RDLONGB	1500.	100.	0	H	N/A	N/A
RNGDEP	300.	25.	0	H	N/A	N/A
SINEX	1000.	25.	2	H	1.	0.
VERTSEA	300.	25.	0	V	N/A	N/A
VERTMIX	100.	10.	0	V	N/A	N/A
WEDGE	1000.	25.	0	H	N/A	N/A

[a]Antenna Pattern: 0=Omni-directional; 1=Gaussian; 2=Sine(X)/X; 3=Cosecant-squared; 4=Generic height-finder.

The external implementation data element requirements that must be specified for each test are listed in Table 4-9.

Table 4-9 External Implementation Data Element Requirements

Test Name	*ef.lerr6*	*ef.lerr12*	*vnp.nrout*	*vnp.nzout*	*vnp.rmax* meters	*vnp.hmin* meters	*vnp.hmax* meters	*vnp.propang* deg
BLOCK	.true.	.true.	1	20	50000.	0.	1000.	0.
COSEC2	.true.	.true.	1	20	50000.	0.	2000.	0.
EDUCT	.true.	.true.	1	20	50000.	0.	200.	0.
GAUSS	.true.	.true.	1	20	50000.	0.	2000.	0.
HILL	.true.	.true.	1	20	50000.	0.	1000.	0.
HORZ	.true.	.true.	1	20	50000.	0.	2000.	0.
HTFIND	.true.	.true.	1	20	50000.	0.	2000.	0.
MIDEL	.true.	.true.	1	20	50000.	0.	5000.	0.
RDLONGB	.true.	.true.	1	20	100000.	0.	1000.	0.
RNGDEP	.true.	.true.	1	20	250000.	0.	1000.	0.
SINEX	.true.	.true.	1	20	50000.	0.	2000.	0.
VERTSEA	.true.	.true.	1	20	300000.	0.	1000.	0.
VERTMIX	.true.	.true.	1	20	50000.	0.	100.	0.
WEDGE	.true.	.true.	1	20	100000.	0.	1000.	0.

The external terrain data element requirements are listed in Table 4-10. Terrain profiles used for specific test cases are listed in Table 4-11 through Table 4-15.

Table 4-10 External Terrain Data Element Requirements

Test Name	*tr.terx* Table	*tr.tery* Table	*tr.itp*	*tr.igr*	*tr.igrnd* Table
BLOCK	4-11	4-11	6	N/A	N/A
COSEC2	N/A	N/A	N/A	N/A	N/A
EDUCT	N/A	N/A	N/A	N/A	N/A
GAUSS	N/A	N/A	N/A	N/A	N/A
HILL	4-12	4-12	67	N/A	N/A
HORZ	N/A	N/A	N/A	N/A	N/A
HTFIND	N/A	N/A	N/A	N/A	N/A
MIDEL	N/A	N/A	N/A	N/A	N/A
RDLONGB	4-13	4-13	167	N/A	N/A
RNGDEP	N/A	N/A	N/A	N/A	N/A
SINEX	N/A	N/A	N/A	N/A	N/A
VERTSEA	N/A	N/A	N/A	N/A	N/A
VERTMIX	4-14	4-14	2	2	4-14
WEDGE	4-15	4-15	5	N/A	N/A

Table 4-11 Terrain Profile for Test Case BLOCK

i	*tr.terx(i)* meters	*tr.tery(i)* meters
1	0.	0.
2	22500.	0.
3	22500.	200.
4	27500.	200.
5	27500.	0.
6	50000.	0.

Table 4-12 Terrain Profile for Test Case HILL

i	*tr.terx(i)* meters	*tr.tery(i)* meters	*i*	*tr.terx(i)* meters	*tr.tery(i)* meters	*i*	*tr.terx(i)* meters	*tr.tery(i)* meters
1	0.	0.0	24	22500.	342.7	47	28250.	312.8
2	17000.	0.0	25	22750.	351.2	48	28500.	295.2
3	17250.	3.6	26	23000.	359.7	49	28750.	277.6
4	17500.	18.0	27	23250.	368.2	50	29000.	260.1
5	17750.	32.4	28	23500.	375.1	51	29250.	242.5
6	18000.	46.7	29	23750.	377.5	52	29500.	224.9
7	18250.	61.1	30	24000.	379.9	53	29750.	206.8
8	18500.	75.5	31	24250.	382.4	54	30000.	187.3
9	18750.	89.9	32	24500.	384.8	55	30250.	167.8
10	19000.	109.4	33	24750.	387.3	56	30500.	148.4
11	19250.	128.9	34	25000.	389.7	57	30750.	128.8
12	19500.	148.4	35	25250.	387.3	58	31000.	109.4
13	19750.	167.9	36	25500.	384.8	59	31250.	89.9
14	20000.	187.3	37	25750.	382.4	60	31500.	75.5
15	20250.	206.8	38	26000.	379.9	61	31750.	61.1
16	20500.	224.9	39	26250.	377.5	62	32000.	46.7
17	20750.	242.5	40	26500.	375.1	63	32250.	32.3
18	21000.	260.1	41	26750.	368.1	64	32500.	17.9
19	21250.	277.6	42	27000.	359.7	65	32750.	3.6
20	21500.	295.2	43	27250.	351.2	66	33000.	0.0
21	21750.	312.8	44	27500.	342.7	67	50000.	0.0
22	22000.	325.8	45	27750.	334.3			
23	22250.	334.3	46	28000.	325.8			

Table 4-13 Terrain Profile for Test Case RDLONGB

i	*tr.terx(i)* meters	*tr.tery(i)* meters	*i*	*tr.terx(i)* meters	*tr.tery(i)* meters	*i*	*tr.terx(i)* meters	*tr.tery(i)* meters
1	0.0	8.0	57	20100.	22.0	113	79200.	184.0
2	300.	8.0	58	20400.	23.0	114	79500.	226.0
3	600.	9.0	59	20700.	24.0	115	79800.	152.0
4	900.	9.0	60	21000.	24.0	116	80100.	201.0
5	1200.	10.0	61	21300.	25.0	117	80400.	244.0
6	1500.	11.0	62	21600.	26.0	118	80700.	152.0
7	1800.	12.0	63	21900.	27.0	119	81000.	143.0
8	2100.	13.0	64	22200.	27.0	120	81300.	91.0
9	2400.	14.0	65	22500.	28.0	121	81600.	107.0
10	2700.	15.0	66	22800.	29.0	122	81900.	152.0
11	3000.	17.0	67	23400.	29.0	123	82200.	152.0
12	3300.	19.0	68	23700.	30.0	124	82500.	170.0
13	3600.	21.0	69	24600.	30.0	125	82800.	152.0
14	3900.	23.0	70	24900.	32.0	126	83100.	66.0
15	4200.	25.0	71	25200.	34.0	127	83400.	70.0
16	4500.	27.0	72	25500.	38.0	128	83700.	121.0
17	4800.	28.0	73	26100.	38.0	129	84000.	152.0
18	5100.	30.0	74	26400.	36.0	130	84300.	170.0
19	5400.	31.0	75	26700.	34.0	131	84600.	141.0
20	5700.	31.0	76	27000.	32.0	132	84900.	139.0
21	6000.	29.0	77	27300.	27.0	133	85200.	147.0
22	6300.	23.0	78	27600.	15.0	134	85500.	177.0
23	6600.	14.0	79	27900.	6.0	135	85800.	152.0
24	6900.	9.0	80	28200.	1.0	136	86100.	61.0
25	7200.	7.0	81	28500.	0.0	137	86700.	61.0
26	7500.	7.0	82	64500.	0.0	138	87000.	70.0
27	7800.	9.0	83	64800.	8.0	139	87300.	44.0
28	8100.	11.0	84	65100.	30.0	140	87600.	11.0
29	8400.	14.0	85	65400.	39.0	141	87900.	1.0
30	8700.	13.0	86	65700.	61.0	142	89400.	1.0
31	9300.	13.0	87	66600.	61.0	143	89700.	61.0
32	9600.	12.0	88	66900.	24.0	144	90000.	84.0
33	9900.	11.0	89	67200.	14.0	145	90300.	152.0
34	10200.	8.0	90	67500.	26.0	146	90600.	152.0
35	10800.	8.0	91	67800.	16.0	147	90900.	101.0
36	11100.	7.0	92	68100.	1.0	148	91200.	40.0
37	12600.	7.0	93	68400.	1.0	149	91500.	15.0
38	12900.	6.0	94	68700.	0.0	150	91800.	20.0
39	14400.	6.0	95	73800.	0.0	151	92100.	2.0
40	14700.	7.0	96	74100.	1.0	152	92400.	10.0
41	15000.	8.0	97	74400.	1.0	153	92700.	4.0
42	15300.	8.0	98	74700.	10.0	154	93000.	1.0
43	15600.	9.0	99	75000.	8.0	155	93300	1.0
44	15900.	10.0	100	75300.	39.0	156	93600.	0.0
45	16200.	11.0	101	75600.	45.0	157	93900.	1.0
46	16500.	11.0	102	75900.	53.0	158	96300.	1.0
47	16800.	12.0	103	76200.	61.0	159	96600.	0.0
48	17400.	12.0	104	76500.	61.0	160	96900.	1.0
49	17700.	13.0	105	76800.	82.0	161	97500.	1.0
50	18000.	13.0	106	77100.	61.0	162	97800.	2.0
51	18300.	14.0	107	77400.	78.0	163	98100.	3.0
52	18600.	15.0	108	77700.	61.0	164	99300.	3.0
53	18900.	16.0	109	78000.	129.0	165	99600.	2.0
54	19200.	18.0	110	78300.	30.0	166	99900.	2.0
55	19500.	20.0	111	78600.	46.0	167	100200.	1.0
56	19800.	21.0	112	78900.	159.0			

Table 4-14 Terrain Profile for Test Case VERTMIX

i	*tr.terx(i)* meters	*tr.tery(i)* meters	*igr*	*igrnd(i)* Note a	*rgrnd(i)* meters
1	0.	0.	1	4	0.
2	50000.	0.	2	0	25000.

[a] Ground composition type: 0=sea water; 1=fresh water; 2=wet ground; 3=medium dry ground; 4=very dry ground; 5=user-defined permittivity and conductivity.

Table 4-15 Terrain Profile for Test Case WEDGE

i	*tr.terx(i)* meters	*tr.tery(i)* meters
1	0.	0.
2	45000.	0.
3	50000.	200.
4	55000.	0.
5	100000.	0.

4.4 Expected Test Results

The expected test result propagation loss versus height values for each of the 14 test cases are listed in tabular form within Table 4-16 through Table 4-29.

Table 4-16 Expected Output for BLOCK Test

Height meters	Propagation Loss dB
50.0	173.5
100.0	170.1
150.0	166.9
200.0	162.3
250.0	157.0
300.0	151.3
350.0	145.8
400.0	140.3
450.0	135.0
500.0	129.6
550.0	124.3
600.0	120.5
650.0	120.9
700.0	130.8
750.0	157.9
800.0	125.0
850.0	120.7
900.0	119.9
950.0	121.7
1000.0	128.3

Table 4-17 Expected Output for COSEC2 Test

Height meters	Propagation Loss dB
100.0	134.4
200.0	124.1
300.0	122.2
400.0	129.6
500.0	126.5
600.0	123.4
700.0	128.0
800.0	126.8
900.0	125.7
1000.0	126.4
1100.0	127.0
1200.0	127.5
1300.0	128.8
1400.0	129.5
1500.0	129.6
1600.0	130.9
1700.0	131.4
1800.0	131.3
1900.0	132.6
2000.0	133.0

Table 4-18 Expected Output for EDUCT Test

Height meters	Propagation Loss dB
10.0	142.7
20.0	147.3
30.0	150.0
40.0	152.2
50.0	155.7
60.0	158.4
70.0	154.3
80.0	149.6
90.0	146.4
100.0	144.2
110.0	143.0
120.0	142.7
130.0	143.2
140.0	145.1
150.0	149.5
160.0	161.4
170.0	151.9
180.0	145.1
190.0	142.3
200.0	141.5

Table 4-19 Expected Output for GAUSS Test

Height meters	Propagation Loss dB
100.0	133.6
200.0	123.4
300.0	121.6
400.0	130.6
500.0	127.0
600.0	124.0
700.0	132.9
800.0	132.2
900.0	129.6
1000.0	139.1
1100.0	139.9
1200.0	138.1
1300.0	148.2
1400.0	150.4
1500.0	149.4
1600.0	160.3
1700.0	163.7
1800.0	163.6
1900.0	175.1
2000.0	179.9

Table 4-20 Expected Output for HILL Test

Height meters	Propagation Loss dB
50.0	195.3
100.0	188.6
150.0	183.6
200.0	179.9
250.0	176.9
300.0	173.6
350.0	169.9
400.0	166.3
450.0	162.8
500.0	159.2
550.0	155.6
600.0	152.1
650.0	148.4
700.0	144.6
750.0	140.3
800.0	135.5
850.0	130.2
900.0	125.2
950.0	121.9
1000.0	124.2

Table 4-21 Expected Output for HORZ Test

Height meters	Propagation Loss dB
100.0	133.6
200.0	123.2
300.0	121.1
400.0	129.6
500.0	124.9
600.0	120.5
700.0	128.1
800.0	125.3
900.0	120.4
1000.0	127.7
1100.0	125.5
1200.0	120.4
1300.0	127.5
1400.0	125.6
1500.0	120.4
1600.0	127.3
1700.0	125.7
1800.0	120.4
1900.0	127.2
2000.0	125.8

Table 4-22 Expected Output for HTFIND Test

Height meters	Propagation Loss dB
100.0	133.6
200.0	123.4
300.0	121.4
400.0	130.0
500.0	125.9
600.0	122.3
700.0	128.6
800.0	126.9
900.0	124.6
1000.0	126.8
1100.0	126.6
1200.0	126.1
1300.0	126.5
1400.0	126.5
1500.0	126.1
1600.0	126.5
1700.0	126.5
1800.0	126.1
1900.0	126.5
2000.0	126.5

Table 4-23 Expected Output for MIDEL Test

Height meters	Propagation Loss dB
250.0	221.9
500.0	220.0
750.0	225.8
1000.0	220.7
1250.0	219.8
1500.0	222.9
1750.0	219.5
2000.0	218.2
2250.0	206.7
2500.0	190.9
2750.0	175.8
3000.0	162.9
3250.0	152.0
3500.0	143.0
3750.0	135.9
4000.0	130.8
4250.0	127.7
4500.0	126.4
4750.0	127.2
5000.0	129.8

Table 4-24 Expected Output for RDLONGB Test

Height meters	Propagation Loss dB
50.0	182.6
100.0	172.0
150.0	171.2
200.0	168.0
250.0	165.2
300.0	161.7
350.0	155.1
400.0	146.7
450.0	140.3
500.0	137.4
550.0	136.9
600.0	138.7
650.0	134.0
700.0	131.5
750.0	135.4
800.0	136.7
850.0	129.9
900.0	142.0
950.0	131.1
1000.0	132.6

Table 4-25 Expected Output for RNGDEP Test

Height meters	Propagation Loss dB
50.0	180.1
100.0	186.5
150.0	186.5
200.0	177.3
250.0	167.3
300.0	155.0
350.0	150.1
400.0	139.4
450.0	123.6
500.0	117.4
550.0	124.7
600.0	118.5
650.0	119.6
700.0	123.6
750.0	127.0
800.0	128.6
850.0	130.0
900.0	131.5
950.0	133.2
1000.0	135.0

Table 4-26 Expected Output for SINEX Test

Height meters	Propagation Loss dB
100.0	133.6
200.0	123.4
300.0	121.6
400.0	130.6
500.0	127.0
600.0	124.0
700.0	133.2
800.0	133.0
900.0	131.8
1000.0	142.4
1100.0	151.9
1200.0	151.7
1300.0	158.4
1400.0	156.0
1500.0	150.8
1600.0	157.8
1700.0	156.2
1800.0	150.9
1900.0	157.7
2000.0	156.3

Table 4-27 Expected Output for VERTSEA Test

Height meters	Propagation Loss dB
50.0	123.5
100.0	134.5
150.0	124.9
200.0	134.0
250.0	136.1
300.0	136.3
350.0	144.4
400.0	150.0
450.0	147.7
500.0	145.0
550.0	143.6
600.0	143.0
650.0	142.8
700.0	142.7
750.0	143.0
800.0	143.3
850.0	143.7
900.0	144.3
950.0	144.7
1000.0	145.3

Table 4-28 Expected Output for VERTMIX Test

Height meters	Propagation Loss dB
5.0	152.9
10.0	153.5
15.0	152.7
20.0	151.0
25.0	149.1
30.0	147.3
35.0	145.8
40.0	144.4
45.0	143.2
50.0	142.1
55.0	141.2
60.0	140.3
65.0	139.5
70.0	138.7
75.0	138.0
80.0	137.4
85.0	136.7
90.0	136.2
95.0	135.6
100.0	135.1

Table 4-29 Expected Output for WEDGE Test

Height meters	Propagation Loss dB
50.0	157.6
100.0	156.5
150.0	156.0
200.0	155.1
250.0	154.3
300.0	154.2
350.0	154.4
400.0	153.0
450.0	149.6
500.0	146.6
550.0	144.2
600.0	141.2
650.0	137.1
700.0	132.9
750.0	129.3
800.0	126.6
850.0	126.0
900.0	128.0
950.0	127.7
1000.0	129.6

4.5 Criteria for Evaluating Results

The calculated propagation loss in dB should match the numerical values in each table at each of the 20 levels shown to within 0.1 dB (1 cB). TPEM rounds its output loss values to the nearest 1 cB, and hence it is possible for differences of 1 cB to exist between different implementations of TPEM. It is expected, however, that in most cases the values will match those in Table 4-16 through Table 4-29 exactly.

4.6 Test Procedure

1. The tester should insure the implementation constants of Table 3-1, are represented within the FORTRAN include files TPEM.INC and FFTSIZ.INC. The TPEM CSCI, along with the driver program MAIN.FOR, is then compiled for execution.
2. An input data file has been provided, as a text file, for each test case.
3. The TPEM CSCI is executed in a form that reads the input data file, performs the calculations, and writes the output to a text file.
4. The output file is compared to the final expected test results to determine satisfactory performance.

4.7 Assumptions and Constraints

Input data elements are assumed to be constrained by the limits listed within Tables 3.3-1 through 3.3-5 of the Software Requirements Specification.

5. REQUIREMENTS TRACEABILITY

(a) The provided driver program that accesses the TPEM CSCI will create an output file for each test case called MAIN.OUT. This output file contains height in meters and corresponding propagation loss in dB that should correspond to the entries in Table 4-16 through Table 4-29 for each test case.

(b) The provided program MAIN.FOR, when compiled with the TPEM CSCI, will read the provided input files containing all necessary information for each test case. Each input file is named for each test case, with a ".IN" extension.

6. NOTES

Table 6-1 is a glossary of acronyms and abbreviations used within this document.

Table 6-1 Acronyms and Abbreviations

Term	Definition
cB	centibel
CSCI	Computer Software Configuration Item
dB	decibel
ef.lerr6	controlling logical flag for error 6
ef.lerr12	controlling logical flag for error 12
EM	electromagnetic
FORTRAN	Formula Translation
km	kilometers
m	meters
maxpts	maximum size of field arrays for all possible applications of TPEM
mxlvls	maximum number of profile levels for all possible applications of TPEM
mxnfft	maximum power of 2 for field array size for all possible applications of TPEM
mxnprof	maximum number of refractivity profiles for all possible applications of TPEM
mxrout	maximum number of range output points for all possible applications of TPEM

Table 6-1 Acronyms and Abbreviations (cont'd)

Term	Definition
mxter	maximum number of terrain profile points for all possible applications of TPEM
mxzout	maximum number of height output points for all possible applications of TPEM
N/A	not applicable
rf.hmsl	refractivity profile height array
rf.lvlep	number of levels in refractivity profiles for particular application of TPEM
rf.nprof	number of refractivity profiles for particular application of TPEM
rf.refmsl	refractivity profile M-unit array
rf.rngprof	refractivity profile range array
sv.antht	antenna height
sv.bwidth	antenna vertical beam width (degrees)
sv.elev	antenna elevation angle (degrees)
sv.freq	EM system frequency (MHz)
sv.ipat	antenna pattern
sv.polar	antenna polarization
TESS	Tactical Environmental Support System
TPEM	Terrain Parabolic Equation Model
tr.igr	number of ground composition types for particular application of TPEM
tr.igrnd	ground composition type array
tr.itp	number of terrain points for particular application of TPEM

Table 6-1 Acronyms and Abbreviations (cont'd)

Term	Definition
tr.rgrnd	round composition type range array
tr.terx	terrain profile range array
tr.tery	terrain profile height array
vnp.hmax	Maximum height output for a particular application of TPEM.
vnp.hmin	Minimum height output for a particular application of TPEM.
vnp.nrout	Number of range output points for a particular application of TPEM.
vnp.nzout	Number of height output points for a particular application of TPEM.
vnp.propang	Maximum propagation angle for particular application of TPEM.
vnp.rmax	Maximum range output for a particular application of TPEM.

7. SAMPLE PROGRAM LISTING

The sample driver program MAIN.FOR, accesses the TPEM CSCI and is provided below.

```
c This is a sample driver program for TPEM routines PEINIT and PESTEP.
c All numeric parameters passed to PEINIT and PESTEP must be in metric
c units.

      program main

      include 'tpem.inc'

      record / errorflag / ef
      record / inputvar / vnp
      record / refractivity / rf
      record / systemvar / sv
      record / terrain / tr

      integer*2 mloss(mxzout) !MLOSS must be declared an INTEGER*2 array
                              !of size at least MXZOUT.

      character*20 filename

      write(*,'(a\)')' Name of input file? '
      read(*, '(a)' ) filename

      open(14, file=filename)

      read( 14, * ) ef.lerr6
      read( 14, * ) ef.lerr12

      read( 14, * ) sv.freq         !Frequency in MHz.
      read( 14, * ) sv.antht        !antenna height.
      read( 14, * ) sv.ipat         !antenna type
      read( 14, '(a1)' ) sv.polar   !antenna polarization.
      read( 14, * ) sv.bwidth       !This value is ignored for Omni antenna.
      read( 14, * ) sv.elev         !This value is ignored for Omni antenna.

      read( 14, * ) vnp.hmin        !Minimum height in m
      read( 14, * ) vnp.hmax        !Maximum output height in m
      read( 14, * ) vnp.rmax        !Maximum output range in m
      read( 14, * ) vnp.nzout       !Number of output height points.
      read( 14, * ) vnp.nrout       !Number of output range points.
      read( 14, * ) vnp.propang     !Maximum propagation angle

      read( 14, * ) rf.nprof        !Number of refractivity profiles
      do i = 1, rf.nprof
         read( 14, * ) rf.rngprof(i)  !Range of profiles in m
      end do
      read( 14, * ) rf.lvlep        !Number of levels in refractivity profiles.
      do j = 1, rf.nprof
```

```
         do i = 1, rf.lvlep
            read(14,*) rf.hmsl(i,j), rf.refmsl(i,j)
         end do
      end do

      read(14,*) tr.igr             ! Number of ground composition types
      do i = 1, tr.igr
         read(14,*) tr.igrnd(i)     ! Ground composition types
         read(14,*) tr.rgrnd(i)     ! Apply at ranges
      end do

      read(14,*) tr.itp             !Number of terrain range/height points
      do i = 1, tr.itp
         read(14,*) tr.terx(i), tr.tery(i)
      end do

c Variables in CAPS are returned.

      call peinit( ef, vnp, rf, sv, tr, HMINTER, ROUT, IERROR )

      if( ierror .ne. 0 ) then
         write(*,*)'*********** ERROR IN PEINIT **************'
         write(*,*)'******** IERROR = ', ierror,' **********'
         return
      end if

      nr = vnp.nrout
      dz = (vnp.hmax-vnp.hmin) / float( vnp.nzout ) !Determine height
                                                    !increment of
                                                    !output points.
      open( 15, file='main.out' )
      write(15,'(1x, a)') '        m            dB     '

      do i = 1, nr

         call pestep( hminter, vnp, rf, tr, sv, ROUT, MLOSS, JSTART,
     +                JEND )

         write(*,*)'range in km = ', rout*1.e-3  !Output to screen

c Recall that MLOSS is the propagation loss in centibels, i.e.,
c MLOSS() = NINT( propagation loss in dB * 10. ).  JSTART = start of
c valid loss points, JEND = end of valid loss points.

         do j = jstart, jend
            write(15,*) j*dz, mloss(j)*.1
         end do

      end do

      close(15)
      end
```

8. INPUT FILE LISTINGS FOR TEST CASES

The input files that are read from the sample driver program MAIN.FOR for each of the 14 test cases are listed below in Sections 8.1 through 8.14. For each test case, an output file containing height in meters and propagation loss in dB (corresponding to Table 4-16 through Table 4-29) is created called MAIN.OUT.

8.1 BLOCK.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
1000.    : Frequency in MHz
25.      : Antenna height in m
0        : Omni antenna type
H        : Horizontal polarization
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
1000.    : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
0        : Range of first refractivity profiles in m
2        : Number of levels in refractivity profiles
0.     0.      : Height & M-unit value of ref. profile 1, level 1
1000. 118.     : Height & M-unit value of ref. profile 1, level 2
1        : Number of ground composition types
0        : Ground composition types (0= sea water, perfect cond. if H Pol)
0.       : Range at which ground composition type is applied in m
6        : Number of terrain range/height points (if 0, then will ignore remainder)
0.        0.   : Range & height of terrain point 1
22500.    0.   : Range & height of terrain point 2
22500.    200. : Range & height of terrain point 3
27500.    200. : Range & height of terrain point 4
27500.    0.   : Range & height of terrain point 5
50000.    0.   : Range & height of terrain point 6
```

8.2 COSEC2.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
1000.    : Frequency in MHz
25.      : Antenna height in m
3        : Cosecant-squared antenna type
H        : Horizontal polarization
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
2000.    : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
```

```
0         : Range of first refractivity profiles in m
2         : Number of levels in refractivity profiles
0.         0.        : Height & M-unit value of ref. profile 1, level 1
1000.     118.       : Height & M-unit value of ref. profile 1, level 2
1         : Number of ground composition types
0         : Ground composition types (0= sea water)
0.        : Range at which ground composition type is applied in m
0         : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.3 EDUCT.IN

```
.true.    : EF.LERR6 error flag
.true.    : EF.LERR12 error flag
10000.    : Frequency in MHz
15.       : Antenna height in m
1         : Gaussian antenna type
H         : Horizontal polarization
5.        : Beamwidth in deg (this value is ignored for OMNI antenna)
0.        : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.        : Minimum output height in m
200.      : Maximum output height in m
50000.    : Maximum output range in m
20        : Number of output height points
1         : Number of output range points
0         : Maximum propagation angle (0=automatic internal calculation)
1         : Number of refractivity profiles
0         : Range of first refractivity profiles in m
21        : Number of levels in refractivity profiles
0.         339.      : Height & M-unit value of ref. profile 1, level 1
.040       335.10    : Height & M-unit value of ref. profile 1, level 2
.1         333.66    : Height & M-unit value of ref. profile 1, level 3
.2         332.6     : Height & M-unit value of ref. profile 1, level 4
.398       331.54    : Height & M-unit value of ref. profile 1, level 5
.794       330.51    : Height & M-unit value of ref. profile 1, level 6
1.585      329.53    : Height & M-unit value of ref. profile 1, level 7
3.162      328.65    : Height & M-unit value of ref. profile 1, level 8
6.310      327.96    : Height & M-unit value of ref. profile 1, level 9
12.589     327.68    : Height & M-unit value of ref. profile 1, level 10
14.        327.67    : Height & M-unit value of ref. profile 1, level 11
25.119     328.13    : Height & M-unit value of ref. profile 1, level 12
39.811     329.25    : Height & M-unit value of ref. profile 1, level 13
50.119     330.18    : Height & M-unit value of ref. profile 1, level 14
63.096     331.44    : Height & M-unit value of ref. profile 1, level 15
79.433     333.12    : Height & M-unit value of ref. profile 1, level 16
100.       335.33    : Height & M-unit value of ref. profile 1, level 17
125.893    338.2     : Height & M-unit value of ref. profile 1, level 18
158.489    341.92    : Height & M-unit value of ref. profile 1, level 19
199.526    346.69    : Height & M-unit value of ref. profile 1, level 20
209.526    347.87    : Height & M-unit value of ref. profile 1, level 21
1         : Number of ground composition types
0         : Ground composition types (0= sea water)
0.        : Range at which ground composition type is applied in m
0         : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.4 GAUSS.IN

```
.true.    : EF.LERR6 error flag
.true.    : EF.LERR12 error flag
1000.     : Frequency in MHz
25.       : Antenna height in m
1         : Gaussian antenna type
H         : Horizontal polarization
```

```
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
2000.    : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
0        : Range of first refractivity profiles in m
2        : Number of levels in refractivity profiles
0.        0.      : Height & M-unit value of ref. profile 1, level 1
1000.    118.     : Height & M-unit value of ref. profile 1, level 2
1        : Number of ground composition types
0        : Ground composition types (0= sea water)
0.       : Range at which ground composition type is applied in m
0        : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.5 HILL.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
1000.    : Frequency in MHz
25.      : Antenna height in m
0        : Omni antenna type
H        : Horizontal polarization
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
1000.    : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
0        : Range of first refractivity profiles in m
2        : Number of levels in refractivity profiles
0.        0.      : Height & M-unit value of ref. profile 1, level 1
1000.    118.     : Height & M-unit value of ref. profile 1, level 2
1        : Number of ground composition types
0        : Ground composition types (0= sea water, perfect cond. if H Pol)
0.       : Range at which ground composition type is applied in m
67        : Number of terrain range/height points (if 0, then will ignore remainder)
0.0      0.0        : Range & height of terrain point 1 in meters
17000.   0.0
17250.0 3.6
17500.   18.
17750.0 32.4
18000.0 46.7
18250.0 61.1
18500.0 75.5
18750.0 89.9
19000.0 109.4        : Range & height of terrain point 10 in meters
19250.0 128.9
19500.0 148.4
19750.   167.9
20000.0 187.3
20250.   206.8
20500.0 224.9
20750.0 242.5
21000.0 260.1
21250.0 277.6
21500.0 295.2        : Range & height of terrain point 20 in meters
21750.   312.8
22000.0 325.8
```

```
22250.0 334.3
22500.0 342.7
22750.0 351.2
23000.0 359.7
23250.  368.2
23500.0 375.1
23750.  377.5
24000.0 379.9       : Range & height of terrain point 30 in meters
24250.0 382.4
24500.  384.8
24750.  387.3
25000.0 389.7
25250.  387.3
25500.0 384.8
25750.0 382.4
26000.0 379.9
26250.0 377.5
26500.0 375.1       : Range & height of terrain point 40 in meters
26750.0 368.1
27000.0 359.7
27250.  351.2
27500.  342.7
27750.0 334.3
28000.  325.8
28250.0 312.8
28500.0 295.2
28750.0 277.6
29000.0 260.1       : Range & height of terrain point 50 in meters
29250.0 242.5
29500.0 224.9
29750.0 206.8
30000.0 187.3
30250.  167.8
30500.0 148.4
30750.  128.8
31000.  109.4
31250.0 89.9
31500.  75.5        : Range & height of terrain point 60 in meters
31750.0 61.1
32000.0 46.7
32250.0 32.3
32500.0 17.9
32750.0 3.6
33000.0 0.0
50000.0 0.0         : Range & height of terrain point 67 in meters
```

8.6 HORZ.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
1000.    : Frequency in MHz
25.      : Antenna height in m
0        : Omni antenna type
H        : Horizontal polarization
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
2000.    : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
0        : Range of first refractivity profiles in m
2        : Number of levels in refractivity profiles
```

```
0.       0.     : Height & M-unit value of ref. profile 1, level 1
1000. 118.      : Height & M-unit value of ref. profile 1, level 2
1        : Number of ground composition types
0        : Ground composition types (0= sea water)
0.       : Range at which ground composition type is applied in m
0        : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.7 HTFIND.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
1000.    : Frequency in MHz
25.      : Antenna height in m
4        : Generic height-finder antenna type
H        : Horizontal polarization
2.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
2000.    : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
0        : Range of first refractivity profiles in m
2        : Number of levels in refractivity profiles
0.        0.       : Height & M-unit value of ref. profile 1, level 1
1000.    118.      : Height & M-unit value of ref. profile 1, level 2
1        : Number of ground composition types
0        : Ground composition types (0= sea water)
0.       : Range at which ground composition type is applied in m
0        : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.8 MIDEL.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
1000.    : Frequency in MHz
10.      : Antenna height in m
1        : Gaussian antenna type
H        : Horizontal polarization
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
5.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
5000.    : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
0        : Range of first refractivity profiles in m
2        : Number of levels in refractivity profiles
0.        0.       : Height & M-unit value of ref. profile 1, level 1
1000.    118.      : Height & M-unit value of ref. profile 1, level 2
1        : Number of ground composition types
0        : Ground composition types (0= sea water)
0.       : Range at which ground composition type is applied in m
0        : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.9 RDLONGB.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
1500.    : Frequency in MHz
100.     : Antenna height in m
0        : Omni antenna type
H        : Horizontal polarization
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
1000.    : Maximum output height in m
100000.  : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Maximum propagation angle (0=automatic internal calculation)
2        : Number of refractivity profiles
0        : Range of first refractivity profile in m
100000.  : Range of second refractivity profile in m
4        : Number of levels in refractivity profiles
0.        0.       : Height & M-unit value of ref. profile 1, level 1
0.        0.       : Height & M-unit value of ref. profile 1, level 2
0.        0.       : Height & M-unit value of ref. profile 1, level 3
1000.    118.      : Height & M-unit value of ref. profile 1, level 4
0.        330.       : Height & M-unit value of ref. profile 2, level 1
191.      352.5      : Height & M-unit value of ref. profile 2, level 2
201.      343.3      : Height & M-unit value of ref. profile 2, level 3
1201.     461.1      : Height & M-unit value of ref. profile 2, level 4
1        : Number of ground composition types
0        : Ground composition types (0= sea water)
0.       : Range at which ground composition type is applied in m
167      : Number of terrain range/height points (if 0, then will ignore remainder)
 0000.      8        : Range & height of terrain point 1 in meters
 0300.      8
 0600.      9
 0900.      9
 1200.     10
 1500.     11
 1800.     12
 2100.     13
 2400.     14
 2700.     15        : Range & height of terrain point 10 in meters
 3000.     17
 3300.     19
 3600.     21
 3900.     23
 4200.     25
 4500.     27
 4800.     28
 5100.     30
 5400.     31
 5700.     31        : Range & height of terrain point 20 in meters
 6000.     29
 6300.     23
 6600.     14
 6900.      9
 7200.      7
 7500.      7
 7800.      9
 8100.     11
 8400.     14
 8700.     13        : Range & height of terrain point 30 in meters
 9300.     13
 9600.     12
 9900.     11
```

```
10200.    8
10800.    8
11100.    7
12600.    7
12900.    6
14400.    6
14700.    7     : Range & height of terrain point 40 in meters
15000.    8
15300.    8
15600.    9
15900.   10
16200.   11
16500.   11
16800.   12
17400.   12
17700.   13
18000.   13     : Range & height of terrain point 50 in meters
18300.   14
18600.   15
18900.   16
19200.   18
19500.   20
19800.   21
20100.   22
20400.   23
20700.   24
21000.   24     : Range & height of terrain point 60 in meters
21300.   25
21600.   26
21900.   27
22200.   27
22500.   28
22800.   29
23400.   29
23700.   30
24600.   30
24900.   32     : Range & height of terrain point 70 in meters
25200.   34
25500.   38
26100.   38
26400.   36
26700.   34
27000.   32
27300.   27
27600.   15
27900.    6
28200.    1     : Range & height of terrain point 80 in meters
28500.    0
64500.    0
64800.    8
65100.   30
65400.   39
65700.   61
66600.   61
66900.   24
67200.   14
67500.   26     : Range & height of terrain point 90 in meters
67800.   16
68100.    1
68400.    1
68700.    0
73800.    0
74100.    1
74400.    1
74700.   10
75000.    8
75300.   39     : Range & height of terrain point 100 in meters
```

75600.	45	
75900.	53	
76200.	61	
76500.	61	
76800.	82	
77100.	61	
77400.	78	
77700.	61	
78000.	129	
78300.	30	: Range & height of terrain point 110 in meters
78600.	46	
78900.	159	
79200.	184	
79500.	226	
79800.	152	
80100.	201	
80400.	244	
80700.	152	
81000.	143	
81300.	91	: Range & height of terrain point 120 in meters
81600.	107	
81900.	152	
82200.	152	
82500.	170	
82800.	152	
83100.	66	
83400.	70	
83700.	121	
84000.	152	
84300.	170	: Range & height of terrain point 130 in meters
84600.	141	
84900.	139	
85200.	147	
85500.	177	
85800.	152	
86100.	61	
86700.	61	
87000.	70	
87300.	44	
87600.	11	: Range & height of terrain point 140 in meters
87900.	1	
89400.	1	
89700.	61	
90000.	84	
90300.	152	
90600.	152	
90900.	101	
91200.	40	
91500.	15	
91800.	20	: Range & height of terrain point 150 in meters
92100.	2	
92400.	10	
92700.	4	
93000.	1	
93300.	1	
93600.	0	
93900.	1	
96300.	1	
96600.	0	
96900.	1	: Range & height of terrain point 160 in meters
97500.	1	
97800.	2	
98100.	3	
99300.	3	
99600.	2	
99900.	2	
100200.	1	: Range & height of terrain point 167 in meters

8.10 RNGDEP.IN

```
.true.  : EF.LERR6 error flag
.true.  : EF.LERR12 error flag
300.    : Frequency in MHz
25.     : Antenna height in m
0       : Omni antenna type
H       : Horizontal polarization
5.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.      : Minimum output height in m
1000.   : Maximum output height in m
250000. : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Maximum propagation angle (0=automatic internal calculation)
2       : Number of refractivity profiles
0       : Range of first refractivity profile in m
250000. : Range of second refractivity profile in m
4       : Number of levels in refractivity profiles
0.    330.    : Height & M-unit value of ref. profile 1, level 1
100   342.5   : Height & M-unit value of ref. profile 1, level 2
230.  312.5   : Height & M-unit value of ref. profile 1, level 3
2000. 517.82  : Height & M-unit value of ref. profile 1, level 4
0.    330.    : Height & M-unit value of ref. profile 2, level 1
600.  405.    : Height & M-unit value of ref. profile 2, level 2
730.  375.    : Height & M-unit value of ref. profile 2, level 3
2000. 522.32  : Height & M-unit value of ref. profile 2, level 4
1       : Number of ground composition types
0       : Ground composition types (0= sea water)
0.      : Range at which ground composition type is applied in m
0       : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.11 SINEX.IN

```
.true.  : EF.LERR6 error flag
.true.  : EF.LERR12 error flag
1000.   : Frequency in MHz
25.     : Antenna height in m
2       : Sin(x)/x antenna type
H       : Horizontal polarization
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.      : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Maximum propagation angle (0=automatic internal calculation)
1       : Number of refractivity profiles
0       : Range of first refractivity profiles in m
2       : Number of levels in refractivity profiles
0.       0.      : Height & M-unit value of ref. profile 1, level 1
1000.   118.     : Height & M-unit value of ref. profile 1, level 2
1       : Number of ground composition types
0       : Ground composition types (0= sea water)
0.      : Range at which ground composition type is applied in m
0       : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.12 VERTSEA.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
300.     : Frequency in MHz
25.      : Antenna height in m
0        : Omni antenna type
V        : Horizontal polarization
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
1000.    : Maximum output height in m
300000.  : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0.       : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
0        : Range of first refractivity profiles in m
4        : Number of levels in refractivity profiles
0.       339.      : Height & M-unit value of ref. profile 1, level 1
250.     368.5     : Height & M-unit value of ref. profile 1, level 2
300.     319.0     : Height & M-unit value of ref. profile 1, level 3
1000.    401.6     : Height & M-unit value of ref. profile 1, level 4
1        : Number of ground composition types
0        : Ground composition types (0= sea water)
0.       : Range at which ground composition type is applied in m
0        : Number of terrain range/height points (if 0, then will ignore remainder)
```

8.13 VERTMIX.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
100.     : Frequency in MHz
10.      : Antenna height in m
0        : Omni antenna type
V        : Horizontal polarization
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.       : Minimum output height in m
100.     : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0.       : Maximum propagation angle (0=automatic internal calculation)
1        : Number of refractivity profiles
0.       : Range of first refractivity profiles in m
2        : Number of levels in refractivity profiles
0.       0.      : Height & M-unit value of ref. profile 1, level 1
1000.    118     : Height & M-unit value of ref. profile 1, level 2
2        : Number of ground composition types
4        : Ground composition type 1 (4=very dry ground)
0.       : Range at which ground composition type 1 is applied in m
0        : Ground composition type 2 (0= sea water)
25000.   : Range at which ground composition type 2 is applied in m
2        : Number of terrain range/height points (if 0, then will ignore remainder)
0.        0.  : Range & height of terrain point 1
50000.    0.  : Range & height of terrain point 2
```

8.14 WEDGE.IN

```
.true.  : EF.LERR6 error flag
.true.  : EF.LERR12 error flag
1000.   : Frequency in MHz
25.     : Antenna height in m
0       : Omni antenna type
H       : Horizontal polarization
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0.      : Minimum output height in m
1000.   : Maximum output height in m
100000. : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Maximum propagation angle (0=automatic internal calculation)
1       : Number of refractivity profiles
0       : Range of first refractivity profiles in m
2       : Number of levels in refractivity profiles
0.       0.      : Height & M-unit value of ref. profile 1, level 1
1000.   118.     : Height & M-unit value of ref. profile 1, level 2
1       : Number of ground composition types
0       : Ground composition types (0= sea water, perfect cond. if H Pol)
0.      : Range at which ground composition type is applied in m
5       : Number of terrain range/height points (if 0, then will ignore remainder)
0.          0.    : Range & height of terrain point 1
45000.      0.    : Range & height of terrain point 2
50000.      200.  : Range & height of terrain point 3
55000.      0.    : Range & height of terrain point 4
100000.     0.    : Range & height of terrain point 5
```

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 1997	Final

4. TITLE AND SUBTITLE

TERRAIN PARABOLIC EQUATION MODEL (TPEM) COMPUTER SOFTWARE CONFIGURATION ITEM (CSCI) DOCUMENTS

5. FUNDING NUMBERS

PE: 0603207N
AN: DN305062
WU: D88–MP67

6. AUTHOR(S)

D. B. Sailors, A. E. Barrios

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Command, Control and Ocean Surveillance Center (NCCOSC)
RDT&E Division
San Diego, California 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

TD 2963

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Space and Naval Warfare Systems Command (SPAWAR)
PMW–185
San Diego, CA 92152–5100

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Approved for public release; distribution is unlimited.	

13. ABSTRACT *(Maximum 200 words)*

This document specifies the functional requirements that are to be met by the Terrain Parabolic Equation Model (TPEM) Computer Software Configuration Item (CSCI). A discussion of the input software requirements is presented together with a general description of the internal structure of the TPEM CSCI as it relates to the CSCI's capability.

14. SUBJECT TERMS		15. NUMBER OF PAGES
Mission area: Communication		292
atmospheric physics	radio-frequency wave propagation	16. PRICE CODE
optics	software requirements	

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

21a. NAME OF RESPONSIBLE INDIVIDUAL	21b. TELEPHONE *(include Area Code)*	21c. OFFICE SYMBOL
A. E. Barrios	(619) 553-1429	Code D883

INITIAL DISTRIBUTION

Code D0012	Patent Counsel	(1)
Code D0271	Archive/Stock	(6)
Code D0274	Library	(2)
Code D0271	D. Richter	(1)
Code D88	J. Richter	(1)
Code D883	A. Barrios	(25)
Code D883	R. Paulus	(1)

Defense Technical Information Center
Fort Belvoir, VA 22060–6218 (4)

NCCOSC Washington Liaison Office
Washington, DC 20363–5100

Center for Naval Analyses
Alexandria, VA 22302–0268

Navy Acquisition, Research and Development
Information Center (NARDIC)
Arlington, VA 22244–5114

GIDEP Operations Center
Corona, CA 91718–8000

Space and Naval Warfare Systems Command
PMW–185
Washington, DC 20363–5100